智能蚁群算法及应用

ZHINENG YIQUN SUANFA JI YINGYONG

吴启迪 汪镭 著

上海科技教育出版社

智能蚁群算法及应用

吴启迪　汪　镭　著

上海科技教育出版社

图书在版编目(CIP)数据

智能蚁群算法及应用/吴启迪，汪镭著. —上海：上海科技教育出版社，2004.4

ISBN 7-5428-3537-8

Ⅰ.智… Ⅱ.①吴…②汪… Ⅲ.智能控制—算法 Ⅳ.TP273

中国版本图书馆 CIP 数据核字(2004)第 029074 号

前　言

在智能自动化研究领域，目前具有群体智能特征的算法研究正受到越来越多的关注。在对何为智能，何为智能控制的讨论学术界尚无统一定论的今天，群体智能，作为智能的一种实现方式，至少体现了智能控制理论的多样性，而这恰恰与人类智能的表现形式是吻合的。

所谓群体智能，是指一种通过大量数目的智能体群来实现的智能方式。作为实现群体智能的每一个个体，它的功能相对于整个问题求解来说是有限的，甚至是极其有限的。每个智能个体在整个智能系统中只能实现总体功能的某个子集，或虽然能构成寻优问题的解答，但往往是非最优的解答。作为智能个体本身，在没有得到智能体群的总体信息反馈的时候，它在解空间中的运动是完全没有规律的。只有在受到其他智能体在解空间中行进方式的影响之后，每个智能个体才能表现出在解空间中具有寻优特征的行进状态。当然，作为智能个体，其定义本身就是相对的，其大小和功能要根据所求解的问题而定。并且，每个智能个体，即使处于合理的寻优进程之中，它的个体动态也并不能保证在每个时刻都具有最佳的寻优收敛特征，其智能寻优方式的实现是通过整个智能体群的总体优化特征来体现的。

作为群体智能的一种典型实现，蚁群算法正在受到学术界的广泛关注。这是一种基于种群寻优的启发式搜索算法，由 M. Dorigo 等人于 1991 年首先提出。它充分利用了生物蚁群能通过个体间简单的信息传递，搜索从蚁穴至食物间最短路径的集体寻优特征，以及该过程与旅行商问题求解之间的相似性，得到了具

有 NP 难度的旅行商问题的最优解答。同时，该算法还被用于求解 Job-Shop 调度问题、二次指派问题以及背包问题等，显示了其适用于组合优化类问题求解的优越特征。

蚁群算法之所以能引起相关领域研究者的注意，是因为该种求解模式能将问题求解的快速性、全局优化特征以及有限时间内答案的合理性结合起来。其中，寻优的快速性是通过正反馈式的信息传递和积累来保证的，而其分布式计算的特征又避免了算法的早熟性收敛，同时，具有贪婪启发式搜索特征的蚁群系统又能在搜索过程的早期就找到可以接受的问题解答。这种优越的问题分布式求解模式经过相关领域研究者的关注和努力，已经在最初的算法模型基础上得到了很大的改进和拓展，并被应用到了包括机器人系统、图像处理、制造系统、车辆路径规划、通信系统、工程设计，以及电力系统在内的多种场合，解决了实际系统中的动态资源配置、运动规划、数据分类等问题。正因为如此，为进一步促进世界范围内相关领域的研究工作，IEEE 进化计算会刊于 2002 年 8 月出版了蚁群优化算法特刊，在其中以蚁群算法的成功应用领域——离散优化问题求解为重点，发表了许多优秀的研究论文，集中体现了蚁群算法研究和应用领域的成功进展。

作者是于上世纪末开始注意到这种算法的，并且以此为背景，带领课题组申请获得了三项国家自然科学基金及其他相关高科技研究项目。正是以这些项目为背景，我们才有机会对此种算法从基本结构、算法特点、改进方法、突破途径、实现模式及应用模式等方面做一些系统的研究工作，也才能有相关论文的发表和录用，同

时才能有今天的资料总结和书刊的编辑。在此，本书作者还要特别感谢课题组的康琦、张燕、吴晔、江重光、傅培玉等研究生，他们参与了本书具体综述材料的检索和书稿的整理校对工作；另外还要感谢上海科技教育出版社的同志们，没有他们的努力，本书是不会这么快就呈现在读者面前的。

吴启迪

2004 年 3 月 12 日

本书出版受到国家自然科学基金（60104004，79970030，70271035），上海市科委启明星计划（03QG14053）和国家 973 子项目（2002CB312202）资助。

目　录

第1章　蚁群算法的由来

1.1　群体智能及典型算法实现

受社会性昆虫行为的启发，智能自动化、智能计算等相关领域的研究工作者通过对其行为的模拟，产生了一系列寻优问题求解的新思路，这些研究可被称为针对群体智能的研究。群体智能（Swarm Intelligence）中的群体（Swarm）是指“一组相互之间可以进行直接通信或者间接通信（通过改变局部环境），并且能够合作进行分布问题求解的主体”。而所谓群体智能是指“无智能的主体通过合作表现出智能行为的特性”。这样，群体智能的协作性、分布性、鲁棒性和快速性的特点使之在没有集中控制，并且不提供全局模型的前提下，为寻找复杂的大规模分布式问题的解决方案提供了基础。

群体智能的优点可以描述如下：

（1）群体中相互合作的个体是分布式的，这样的分布模式更适合于网络环境下的工作状态。

（2）系统没有集中的控制指令与数据存储，这样的系统更具有鲁棒性，不会由于某一个或者某几个个体的故障而影响整个问题的求解进程。

（3）系统不通过个体之间的直接通信，而通过非直接通信方式进行信息的传输与合作，这样的系统具有更好的可扩充性，由于系统中个体的增加而增加的通信开销也较小。

（4）系统中每个个体的能力十分简单，每个个体的执行时间也比较短，并且实现较为方便，具有简单性的特点。

由于具有以上这些优点，虽说关于群体智能的研究还处于初

级阶段,并且还存在着许多困难,但是我们可以预言,群体智能的研究代表了智能控制及智能计算研究发展的一个重要方向。

作为群体智能的典型实现,模拟生物蚁群智能寻优能力的蚁群算法和模拟鸟群运动模式的微粒群算法正在受到学术界的广泛关注。其中,蚁群算法由于所提出的时间相对较早,已经得到了广大研究人员较为充分的研究。这也是本书的论述重点。

1.2 基本蚁群算法的起源

蚂蚁是地球上最常见、数量最多的昆虫种类之一,常常成群结队地出现于人类的日常生活环境中。这些昆虫的群体生物智能特征,引起了一些学者的注意。意大利学者 M. Dorigo, V. Maniezzo 等人在观察蚂蚁的觅食习性时发现,蚂蚁总能找到巢穴与食物源之间的最短路径。经研究发现,蚂蚁的这种群体协作功能是通过一种遗留在其来往路径上的叫做信息素(Pheromone)的挥发性化学物质来进行通信和协调的。化学通信是蚂蚁采取的基本信息交流方式之一,在蚂蚁的生活习性中起着重要的作用。通过对蚂蚁觅食行为的研究,他们发现,整个蚁群就是通过这种信息素进行相互协作,形成正反馈,使多个路径上的蚂蚁逐渐聚集到最短的那条路径上来的。

这样,M. Dorigo 等人于 1991 年首先提出了蚁群算法。其主要特点就是:通过正反馈、分布式协作来寻找最优路径。这是一种基于种群寻优的启发式搜索算法。它充分利用了生物蚁群能通过个体间简单的信息传递,搜索从蚁穴至食物间最短路径的集体寻优特征,以及该过程与旅行商问题求解之间的相似性,得到了具有 NP 难度(Non-deterministic Polynomical Completeness)的旅行商问题的最优解答。同时,该算法还被用于求解 Job-Shop 调度问题、二次指派问题,以及背包问题等,显示了其适用于组合优化类问题求解的优越特征。

1992 年,M. Dorigo 在他的博士论文中进一步提出了蚁群系统(Ant System,AS)。在这篇论文中,根据信息素增量的不同计算方法,M. Dorigo 给出了三种不同的模型,分别称之为蚁周、蚁量和蚁密模型;同时通过大量实验,讨论了不同参数对算法性能的影响,确定了算法主要参数的有效区间。

这样,蚁群算法(Ant Colony System, ACS)所表现出来的群体智能就很好地模拟了蚁群做事的流程性及柔性分工特征,并且模拟了蚁群处理工作链脱节和延迟问题所采取的岗位替补与协同模式。

通过多年来世界各地研究工作者对蚁群算法的精心研究和应用开发,该算法现已被大量应用于数据分析、多机器人协作问题求解,以及电力、通信、水利、采矿、化工、建筑、交通等领域。

这里,要解释简单的程序规则如何使蚁群算法完成如此复杂的功能,其答案只能是:简单规则中的智能涌现。事实上,每个蚂蚁智能体并不是像我们想像的那样需要知道整个世界的信息,它们只需要关心很小范围内的局部信息,而且只需根据这些局部信息,利用几条简单的规则来进行决策。这样,在蚁群算法的群体求解模式中,其复杂性的性能特点就会通过群体协作凸显出来。这就是人工生命、复杂性科学的根本规律。

1.3 蚁群个体的运动规则

作为蚁群算法的蚂蚁个体,其运动和通信的简单规则只需包含以下几个主要方面:

(1) 搜索范围:可具体设定蚁群个体的搜索参数半径,这样就限制了其运动过程中的观察能力和移动距离。

(2) 局部环境:蚂蚁个体仅需要感知它周围的局部环境信息,并且该局部环境中的信息素是按一定速度消失的。

(3) 觅食规则:每只蚂蚁只在其能感知的范围内进行信息探索和留存,在局部环境中,哪一点的信息素最多,就以较大的概率

决定了它的运动方向。这样,虽然在其运动过程中,会出现小概率的搜索错误,但从总体上说,其搜寻的效率和正确性会通过其他蚂蚁的行为反馈加以调整。

(4) 移动规则:每只蚂蚁都朝信息素最多的方向移动,当周围没有信息素指引的时候,蚂蚁会按照自己原来运动的方向惯性地运动下去,并且,在运动的方向上有一个随机的小的扰动,以保留原来的运动记忆。如果发现有其已经经过的地点,则以较大概率进行避让。

(5) 避障规则:如果在蚂蚁即将移动的方向上存在障碍物,则它会随机选择另一个方向,或者按照信息素的指引继续其觅食行为。

(6) 通信规则:实际上,每只蚂蚁是通过其信息素的播撒和感知来进行通信的。其具体规则是多元化的,它可以在找到相对最优解的时候散发最多的信息素,并且随着它走的距离越来越远,播撒的信息素越来越少。

1.4 实例说明及应用状况

这里,我们先来看一下蚁群的最短路径搜寻过程。其中,信息素的散发和感知是最重要的。信息素多的地方,经过的蚂蚁就多,经过一段时间的正反馈过程,就会有更多的蚂蚁聚集过来。假设有两条路从蚁穴通向食物。开始的时候,走这两条路的蚂蚁数量同样多(或者在较长的路上蚂蚁多一些,这无关紧要),另外,蚂蚁沿着一条路到达终点以后就马上返回。这样,在短的路径上,蚂蚁来回一次的时间就短,这也意味着其路程重复的频率就快,因而在单位时间里走过的蚂蚁数目就多,撒下的信息素自然也会多,其后自然会有更多的蚂蚁被吸引过来,从而撒下更多的信息素……而长的那条路上情况正好相反。因此,随着时间的推移,越来越多的蚂蚁会聚集到较短的路径上来,这样就近似找到了最短路径。对

于局部最短路径和全局最短路径的问题，即局部最优和全局最优的问题求解模式，可以说，实际上蚂蚁是逐渐接近全局最优的。这源于蚂蚁会按照一定的概率不往信息素高的地方走而另辟蹊径，这可以理解为一种创新。这种创新如果能缩短路途，那么根据刚才叙述的原理，更多的蚂蚁就会被吸引过来。

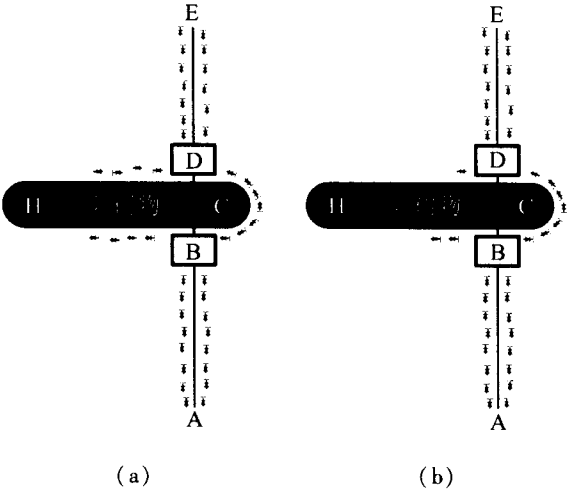


图 1-1 生物蚁群搜索过程示例

图 1-1 具体说明了这个过程，从 A 出发到 E 有两条路径，左边的较长，右边的较短。在这两个图中，开始的时候[图(a)]蚂蚁选择两条路径的机会是均等的，当时间逐渐流逝以后[图(b)]，更多的蚂蚁就会聚集到右边较短的路径上来。

在蚂蚁运动过程中，其之所以具有智能行为，完全归功于它的简单行为规则，而这些规则综合起来又具有多样性和正反馈这两个方面的特点。多样性保证了蚂蚁在觅食的时候不进入无限的死循环，而正反馈机制则保证了相对优良的信息能够被保存下来。这里，我们可以把多样性看成是一种创造能力，而正反馈则是一种

学习强化能力。正反馈的力量也可以被比喻成权威的意见，多样性则是打破权威而体现的创造性，正是这两点的巧妙结合才使得智能行为涌现了出来。

蚁群原先有一条行走的路径[比如从食物 A 到穴 E，反之亦可，见图 1-1(a)]。之后，突然出现了一个障碍物，切断了这条路径。这样，从 A 到 E 的蚂蚁在 B 点(或反方向移动的蚂蚁在 D 点)就必须决定是向左移动还是向右移动[见图 1-1(b)]。在整个决策过程中，受先行蚂蚁所留下的气息痕迹的强度影响，右边路径上气息的较高浓度将给蚂蚁更强烈的刺激。因此，总体上蚂蚁向右转的概率会更大一些。当然，第一只到达 B(或 D)点的蚂蚁向右转或向左转的概率是相等的(这显然是由于在可供选择的两条路径上没有任何历史气息所致)。由于路径 BCD 比 BHD 要短，因此第一只沿此路径前进的蚂蚁将在第一只沿 BHD 前进的蚂蚁之前到达 D。结果，一只从 E 回到 D 的蚂蚁在路径 DCB 上就会发现有较多的气息痕迹，这是由那些随机决定通过 DCBA 绕过障碍物的半数从 E 到 A 方向的蚂蚁，和那些已经通过 BCD 到达 E 的蚂蚁留下的。因此，它们将更倾向于选择(在概率意义上)路径 DCB，而不是 DHB。最终，单位时间通过路径 BCD 的蚂蚁数将比通过 BHD 的蚂蚁数更大。这就使短路径上的气息比长路径上的气息增长更快，故而所有的蚂蚁选择路径的概率将很快地偏向较短的路径。最后，所有蚂蚁都会很迅速地作出反应，选择较短的路径。

以蚁群算法为代表的群体智能已成为当今分布式人工智能研究的一个热点，许多源于蜂群和蚁群模型设计的算法已越来越多地被用于企业的运转模式的研究。美国五角大楼正在资助关于群体智能系统的研究工作——群体战略(Swarm Strategy)，它的一个实战用途是通过运用成群的空中无人驾驶飞行器和地面车辆来转移敌人的注意力，让自己的军队在敌人后方不被察觉地安全行进。

英国电信公司和美国世界通信公司以电子蚂蚁为基础,对新的电信网络管理方法进行了试验。群体智能还被应用于工厂生产计划的制定和运输部门的后勤管理。美国太平洋西南航空公司采用了一种直接源于蚂蚁行为研究成果的运输管理软件,结果每年至少节约了1000万美元费用开支。英国联合利华公司已率先利用群体智能技术改善其一家牙膏厂的运转状况。美国通用汽车公司、法国液气公司、荷兰公路交通部和美国一些移民事务机构也都采用这种技术来改善其运转的机能。鉴于群体智能广阔的应用前景,美国和欧洲联盟均于近几年开始出资资助基于群体智能模拟的相关研究项目,并在一些院校开设群体智能的相关课程。牛津大学出版社1999年出版的E. Bonabeau和M. Dorigo等人编写的专著《群体智能:从自然到人工系统》(*Swarm Intelligence: From Natural to Artificial System*),以及2001年出版的J. Kennedy和R. Eberhart编著的《群体智能》(*Swarm Intelligence*)进一步扩大了群体智能的影响。IEEE进化计算会刊也于2002年8月出版了蚁群优化算法特刊。国内,国家自然科学基金"十五"期间学科交叉类优先资助领域中的认知科学及其信息处理的研究内容中也明确列出了群体智能领域的进化、自适应与现场认知主题。

1.5 蚁群算法的研究及应用领域纵览

由前面的论述可知,蚁群算法是一种启发式算法,它来源于对蚂蚁群体搜索行为的研究。它还充分模拟了实际蚁群寻求最短路径的协作优化特性。由于蚂蚁寻求最短路径的行为类似于旅行商(TSP)问题的解决过程,因而蚁群算法能在TSP问题实例中得到最优的结果。另外,蚁群算法还被用于作业车间调度问题、二次分配问题、多维背包问题、数据的特征聚类过程,并取得了很好的寻优结果。

蚁群算法在解决组合优化类问题求解方面表现出了突出的适

用特征。人工蚁群中的多个智能体通过正反馈确保了最优化过程的快速性，而早熟收敛则可以由蚁群算法的分布式计算特性来加以避免。同时，由于它的贪婪式启发搜索特性，在搜索过程的早期就可以找到可以接受的解。这样，在一种问题求解模式中，同时结合了问题求解的快速性、全局最优特征以及在优化过程初期解的合理性等特性，从而引起了相关领域研究者的注意。

通过相关领域研究者的关注和努力，蚁群算法在最初模式的基础上得到了许多改进和扩展，并在大量领域获得了应用，比如机器人系统、图像处理、制造系统、车辆路径系统、通信系统、工程设计领域及电力系统等。它还被用于各类动态资源分配、行动规划和数据聚类等相关问题的研究中。

在本章中，作者将全面讨论用于离散空间优化的传统蚁群算法。在后续章节中，作者将把蚁群算法引入连续空间优化过程的研究工作作一阶段性总结。

1.5.1 适用于离散空间优化问题的蚁群算法

如前所述，蚁群算法是一种分布式寻优算法，适用于 TSP 问题的求解。在蚁群算法中，包含了一组称为蚂蚁的协作智能体，它们通过一定的策略进行互相合作及间接信息交互，从而搜索 TSP 问题的最优解。在人工蚁群中，被定义的人工蚂蚁使用信息素（或可叫做痕迹）来进行互相的交流。许多论文中的结果表明，蚁群算法要优于其他受自然启发得出的寻优算法。由于人工蚁群的移动类似于实际的蚁群旅行，所以将它用于对称的或不对称的 TSP 问题是可以有直接对应的方法的。在这些应用中，蚁群的正反馈特性有助于快速发现好的解，而贪婪启发式规则有助于在搜索过程的初期找到可以接受的解。

蚁群算法也可被用于作业车间调度问题。在相关文献研究中，提出了用于改善蚁群算法性能的策略；然后将其用于作业车间调度问题，还研究了蚁群算法的领导者和精英策略。另外，还引入

了带延迟的学习思想来改善蚁群算法的性能,并在其中强调了合作策略的作用。

在解决二次指派问题(QAP)时,可以使用一个结合局部搜索策略的混合蚁群算法。它使用信息素来修正二次指派问题的解,而不像传统的蚁群算法那样使用信息素来构造完整的解。在此领域的研究中,可以定义一个与蚁群算法一致的启发式策略。它所具有的一个显著特征是,在每一个构造步中使用了一个新的下界。为了评价基于蚁群算法求解的性能,有人将其用于二次指派问题的几个典型实例,并将所得的结果与其他进化算法和启发式算法相比较。另外,为了解决子集问题,还可将蚁群算法进行修正,用于多维背包问题的典型实例计算研究。

蚁群算法还是一个灵活的特征数据聚类问题的求解工具,可以将蚁群算法与遗传算法结合起来,在选择其表示模式后,选择有效的聚类对象。

所有这些问题解决的过程都是在离散空间中进行的,在离散空间优化问题求解中,蚁群算法表现出了非凡的特性。

1.5.2 蚁群算法的改进

从蚁群算法发表并解决了旅行商问题之日起,它就引起了全世界相关研究领域的广泛关注。在此基础上,研究人员将传统蚁群算法进行了很多改进研究。

最具贪婪式寻优特征的改进蚁群算法为最大最小蚁群系统(Max-Min Ant System,MMAS)。它在几个重要的方面与传统蚁群算法不同,它具有增强的贪婪搜索特性,能有效地使局部搜索朝有希望的搜索空间前进。通过相关实例,可以看到它有比传统蚁群算法更好的寻优特性。

在改进的蚁群算法中,Ant-Q 算法是一种新颖的、分布式的增强型学习算法,它与 Q-learning 算法有很多相似点。在此框架下,传统的蚁群算法可被看作 Ant-Q 的一个特例。在一些实例中,

Ant-Q 算法比传统的蚁群算法性能要好。尤其是对于一些困难的非对称 TSP 问题，Ant-Q 能够找到通常只由专用算法才能找到的解。

将禁忌搜索作为局部搜索策略引入蚁群算法，可以得到一个具有更强计算功能的算法。这种改进的蚁群算法可被用于一些大规模的布局或空间规划问题，并且可以得到全局最优解。在具体的蚁群算法实现中，禁忌搜索可被用于局部改进阶段，产生一个增强型算法。

其他的智能算法模型，如遗传算法、多智能体模型、阶次模型、增强型算法、基于图的一般框架模型、孤岛模型等，也可被引入传统的蚁群算法，以产生积极的改进效果。甚至还有研究者开发了一个基于蚁群模型的细胞神经网络（Cellular Neural Network，CNN），并将其用于电路的振荡和混沌特性的研究。蚁群的协作特征与遗传算法的进化特征可以结合起来，以优化传统蚁群算法的性能。通过引入遗传算法，就可以借助其进化过程来选择蚁群算法的优化参数。在多智能体模型中，可以模仿实际的蚂蚁，在其中可以有两个不同类型的智能体，每一类都有特定的任务，也可以根据问题求解的需要设定多个协作智能体。在考虑噪声的情况下，所定义的智能体可以使用信息素来彼此交流。在基于蚁群算法的组合优化问题研究中，已发现对于几个 TSP 实例，在平均行为方面，修正的蚁群算法可以与其他方法相竞争，并且表现出更好的最差情况特征。所加强的蚁群算法是根据合作机制定义的。基于该算法可以设计一个开关箱断路器，与其他基准例子相比，它表现出较低的计算复杂度。基于图的蚁群算法来源于图的构造概念。相关文献指出，在特定的条件下，这种基于图的蚁群在每一代产生的解可以以任意接近于 1 的概率收敛于给定问题实例的最优解。在孤岛模型中，几群蚂蚁智能体被用于蚁群优化算法，并引入了一个预测函数，用于使蚁群能根据决策后所达到的状态来决定蚁群的

运动。在产生同步振荡行为模式的 CNN 模型中,蚁群的行为模式可被用来作为创建该模型的原型。

其他研究者集中于对蚁群算法的启发式特性、在增强型学习框架下的蚁群建模、蚁群算法的形式描述、群体智能的计算模型等进行研究,并且在相关应用研究中取得了很大进展。

1.5.3 蚁群算法的工程应用

在蚁群算法的工程应用中,由于微型机器人(或移动机器人)的协调策略、行为策略、优化策略和蚁群系统有内在相似性,所以蚁群系统模型在多机器人系统协调、分布式控制规则设计、停滞恢复策略研究和气味传感机器人开发等方面有很大的参考意义。在微型机器人系统动态模型、微型自治机器人系统的群体行为设计、可移动机器人的行为控制系统、痕迹跟踪机器人蚂蚁和自治机器人的导航策略等领域,蚁群算法的思想也被引入其中。这些机器人系统模仿了蚁群系统的群体优化能力,并具有实际蚂蚁系统的协同操作、协同决策、协同优化和分布式通信特性。

在其他工程应用中,如交通系统控制中,蚁群算法还被引入公共汽车路线规划和其他类型的车辆路线规划问题。在通信网络应用中,蚁群智能体借助于运动协调和模拟信息素沉积来解决负载平衡问题。在电力系统应用中,故障单元估计问题可被描述为一个组合优化问题,因而也可以使用蚁群系统来解决。虽然热电经济分配问题被分解为两个子问题,但是合作蚂蚁智能体可以有效地处理这个问题的约束,并给出合理的解。在制造系统控制中,也可以使用蚁群系统的通信和协调能力。在图像处理中,可以使用蚁群算法来揭示大图中的一些结构特征,还可将蚁群算法用于图像着色问题求解。在分布式自治系统的实现中,蚁群系统可用于动态资源配置和协调运动规划。

在工程设计领域还有一些蚁群算法的典型应用。例如,在化工领域中合成路径的设计和在串、并联系统中冗余配置问题的解

决中，都可引入相应的蚁群系统算法。

大多数蚁群算法研究主要是被用于解决离散空间中的优化问题的，但同时，对于连续空间中的优化问题，用蚁群算法也有了一些初步的结论。因为在大量的工程设计领域，设计问题总可以被描述为连续设计空间中的优化问题，因此也有一些文章中对蚁群算法作了相应修正，以使其适用于该领域。但是这些论文中没有给出一个直接适用于连续空间的蚁群算法框架。

关于蚁群算法的研究及应用成果，我们还要在后面的章节中进行详细的分类综述。

1.6 蚁群算法的总体特征及作者的工作

蚁群算法之所以能引起相关领域研究者的注意，是因为这种求解模式能将问题求解的快速性、全局优化特征以及有限时间内答案的合理性结合起来。其中，寻优的快速性是通过正反馈式的信息传递和积累来保证的，而其分布式计算的特征又避免了算法的早熟性收敛，同时，具有贪婪启发式搜索特征的蚁群系统又能在搜索过程的早期就找到可以接受的问题解答。这种优越的问题分布式求解模式经过相关领域研究者的关注和努力，已经在最初的算法模型基础上得到了很大的改进和拓展。

作者在近几年的工作中，将原适用于离散空间组合优化问题求解的蚁群算法的优越寻优模式拓展至连续空间内的线性和非线性函数寻优问题的求解中，证明了蚁群算法在连续空间寻优问题求解中的适用特征。在此基础上，进行了基于蚁群算法的系统参数辨识及智能控制研究，并在算法的计算机仿真实现之后，在国家高性能计算基金支持下进行了算法的并行实现机理研究，这些都为蚁群算法在实际系统中的应用实现开辟了道路。这在本书以后的内容中将进行介绍，并将论述系统的求解模式和应用模式。

第2章　蚁群算法的研究成果

1998年10月14～16日，首届蚁群优化国际会议在比利时布鲁塞尔召开。此次会议的目的是将有着不同兴趣爱好的研究人员和实践者聚拢在一起，研究蚁群优化的本质及实现模式，涉及的学科领域从生物学到物理学、工程学和计算机科学。会议考虑的主题是结构化复杂行为的分析和拟合，包括简单通信智能体的大量合成，其中把蚁群系统作为模型基础，它标志着蚁群算法的研究已经得到了国际上的广泛支持。本章中，我们将对蚁群系统的算法研究成果作一系统综述。

2.1　基本蚁群算法的提出和分析

近年来，不同领域的研究者从适应性角度对生物行为模拟表现出了很大的兴趣。蚁群作为社会性昆虫，在解决无法由单个蚂蚁所能执行的任务时显示了集体行为的作用。在蚁群中，信息素是传递全局行为重要信息的媒介。

1996年，M. Dorigo等于IEEE刊物上发表了一篇奠基性论文，文中对蚁群系统的最基本模型和特点进行了综合分析，并通过实例仿真讨论了该模型的特征和应用。M. Dorigo等还推荐将蚁群算法用于进行组合优化问题求解，并指出该模型的主要特点是正反馈、分布式计算以及具有结构性特征的贪婪式搜索模式。其中，正反馈特征可以解释蚁群寻优的快速性，分布式计算避免了蚁群在寻优空间中的过早收敛，而贪婪式搜索模式则有助于在搜索过程的早期阶段就找到可行的问题解答。论文中应用蚁群算法求解了经典的旅行商(TSP)问题，并给出了仿真结果。另外，还讨论

了算法的参数选择和早期建模问题，并将其与禁忌搜索法和模拟退火法在同一个问题(TSP)求解中进行了性能比较。为了说明该算法的鲁棒性，还讨论了如何将蚂蚁系统(AS)应用于其他的优化问题，如不对称旅行商(ATSP)问题、二次分配问题和作业安排调度等，并对蚁群算法的突出特性，包括全局数据结构校正、分布式通信和蚁群的概率转移特性等进行了分析。

由于蚁群以及大多数社会性的昆虫都是一个分布式的系统，其特点是个体简单，但能表现出很高的组织性和社会性。所以作为一个组织，蚁群能完成一些非常复杂的工作。而这样的工作对于单个蚂蚁来说是非常困难的。在计算机科学研究领域中，研究者对于蚁群的行为模拟和它们的自组织能力非常感兴趣，因为这些能力可以提供一种用于解决复杂的最优化和分布式控制问题的模型。从真实蚁群中可以抽象出来一些相关模型，特别是，已经有论文通过在蚁群筑巢信息传递过程中的应用范例，研究了其分布式自组织行为的模拟方式。

另外，在学习自动机的概念体系下，也可对蚁群算法进行分析。原先的学习自动机模型是用来从生理和心理上描述人的行为的，而有一类相互作用的学习自动机，它能够用来描述蚁群的各种行为能力，例如找到离蚁巢最近的食物、然后返回的能力。蚁群算法就是为模拟蚁群的群体智能行为而设立的。著名的互连式学习自动机，就是用于在分散模式下解决马尔可夫决策问题的标准模型，其模型结构和蚁群算法非常匹配。这样，通过学习自动机的研究，就能非常好地了解蚁群算法为何有效的原因，而且知道蚁群算法在多智能体系统(MAS)中为何能成为一个非常有用的理论工具。为了说明以上观点，有人给出了一个学习自动机直接用到一般马尔可夫游戏问题中的实例。

在蚁群系统设计中，合成信息素的作用是不可忽略的。有人对蚂蚁觅食行为中信息素的作用及蚂蚁社会中综合创造力的本质

进行了系统研究。该研究不仅具有公认的创造力特征，还提供了在不同应用领域中实践意义，并且给出了研究创造力本质和人工创造特性的科学方法。

如果将社会性昆虫的行为引入电脑计算领域，把蚁群本身设想成一个电脑网络，生物学家也可以将其复杂的行为模式进行分解。在此领域也有相关的论文报道，由于非本书重点，故在此不予赘述。

2.2 蚁群算法的改进综述

在基本蚁群算法被提出之后，已被多个领域的研究工作者进行了改进。其中包括最大最小蚁群算法、Ant-Q 算法、带禁忌搜索策略的蚁群算法、多群体蚁群算法、具有变异特征的蚁群算法、自适应蚁群算法、与其他智能算法相结合的蚁群算法等。这里，我们仅对一些效果较好的改进型蚁群算法进行综述。

2.2.1 最大最小蚁群系统

从其名称上就能很好理解该算法的主要特征。通过对传统蚁群优化算法——蚁群系统的研究，显示其作为一种解决离散空间内复杂组合优化问题的方法是可行的。然而，与其他具有规则性的算法相比，蚁群系统在处理很多传统基准问题，如 TSP 问题时，其性能相对来说还不是最好。为了证明蚁群优化算法也能够取代现存算法来处理复杂组合优化问题，最近在这个领域的研究主要集中在算法变量上，使算法变量的性能要比传统蚁群系统的好。有研究者提出了源自于蚁群系统的蚁群优化算法——最大最小蚁群系统(MMAS)。该系统在几个重要的方面与传统蚁群系统有区别，他们的实验研究说明了该算法的有效性。此外，他们还将 MMAS 的特征之一(使用比蚁群算法更贪婪的搜索模式)与论文中涉及的组合优化问题的搜索空间分析结果联系起来，通过对旅行商问题和二次分配问题的计算机仿真，所得结果表明，MMAS 是

现存解决这些问题的算法中性能最好的。

2.2.2 多重蚁群算法

在1991年,A. Colorni 等通过对蚂蚁寻找食物的行为和使用信息素的通信方式类比,提出用蚁群算法解决 TSP 问题。在基于蚁群算法的 TSP 问题求解中,由许多简单的蚂蚁智能体组成蚁群,这些蚂蚁智能体不断地访问各节点城市,希望能找到连接最近城市的路线,并留下很强的信息素,完成旅程的蚂蚁根据距离的不同在它们所经过的路线上留下不同强度的信息素,即在 TSP 巡游中有可能成为最佳路线的就留有更强的信息素。所以蚂蚁智能体通过使用这种正反馈机制,能在搜索空间中找到最优解的存在区域。为了更好地解决 TSP 问题,有论文提出了由传统蚁群算法扩展而来的多重蚁群算法。这种算法由几个蚁群来协同解决 TSP 问题,而传统的蚁群算法中则只有一个蚁群。而且,在蚁群群体层的交互作用中还使用了正负两种信息素效应。由于引入了群体层的交互作用,蚁群就能更好地交换问题解决过程的规划信息,并保持它们在搜索过程中的多样性。这样,在使用几乎相同的智能体策略的条件下,引入群体层交互作用的算法就比传统算法显示了更好的性能。

在多重蚁群算法中,为了求解优化问题,几个蚁群可以相互合作,以找到最佳解答。在某些时间点上,各蚁群可以相互交换好的解答信息。如果交换的信息量不太大,则多重蚁群算法就能在不同的处理器上分配不同的蚁群,从而轻易地实现并行处理。有论文详细研究了在蚁群间具有不同信息交换方式的多重蚁群算法的性能表现,而且比较了多起点单蚁群算法中不同数量的蚁群行为,并以 TSP 问题和二次指派问题作为测试问题。

2.2.3 具有变异特征的蚁群算法

蚁群算法是一种新型的模拟进化算法,初步的研究已经表明该算法具有许多优良的性质。但该算法也存在一些缺点,如计算

时间较长、执行复杂度较高等。为了克服这些缺点，有人给出了一种新的蚁群算法——具有变异特征的蚁群算法。在基本蚁群算法中可引入变异机制，这样就充分利用了二元交换法简洁高效的特点，使得该方法具有较快的收敛速度，达到节省计算时间的目的。已有计算机仿真结果表明，该方法是行之有效的。

2.2.4 自适应蚁群算法

在蚁群算法的自适应特性研究中，一种分布平衡的自适应蚁群算法显示了优越的寻优特征。针对蚁群算法加速收敛和早熟停滞现象之间的矛盾，有人提出了一种分布平衡的自适应蚁群算法，可以在加速收敛和防止早熟、停滞现象之间取得很好的平衡。该算法可以根据优化过程中解的分布平衡度，自适应地调整路径选择概率的确定策略和信息量更新策略。以数种对称和不对称 TSP 问题为例所进行的计算结果表明，该方法比一般蚁群算法具有更好的收敛速度和稳定性，更适合于求解大规模的 TSP 问题。

2.2.5 蚁群算法的收敛性研究

在蚁群优化算法的收敛性证明中，有人证明了一类蚁群算法的收敛性质。特别地，可以证明，在该类算法条件下，对于一个任意小的常量（$\varepsilon>0$）和足够多的算法迭代次数 t，找到最优解的概率至少是 $P^*(t)\geqslant 1-\varepsilon$，而且当 t 趋向于无穷大时，此概率就趋向于 1。还可证明，当找到最优解之后，在有限的循环次数内，与最优解相关的信息素积累将超过任何其他的信息素积累。

还有人发展了一种以结构图为基础的、能够解决组合最优化问题的启发式蚁群算法，并给出了其收敛性证明。在该框架下，结构图被赋予当前所考察优化问题的实例，并合理借用路径的概念加以表达。计算表明，在一定的条件下，基于图的蚁群算法通过反复的迭代能够产生对于给定问题接近于最优的解。

另有论文讨论了蚁群优化（ACO）算法如何保证全局最优。同样，在所构建的基于图的蚁群系统中，分析了各蚁群优化算法变

量,并模拟了一定数量蚂蚁在图中的随机走动,其目的是为了解决图中的最优路径搜索问题。对于一个特定的蚁群优化算法,文中证明了其实现最优解决方案的概率为1。该结果显示,图中的最佳路径能得到随机动态过程的吸引子,所提随机动态过程可通过带参数的蚁群算法加以实现。

2.2.6 新的蚁群算法研究思路

有人基于生物学意义上的蚁群提出了一种新的搜寻算法,并在论文中描述了一个基于原始蚁群寻找食物行为模型(Pachycondyla Apicalis 模型)的新的寻优算法。这些蚁群的特点是,用一个相对简单但更有效率的策略来寻找食物。在这个过程中,每个蚂蚁尽量在它们巢周围一个特定的区域独自搜寻食物。蚁群搜寻行为由一系列并行的局域搜索组成,它们能敏锐地觉察到成功的搜寻位置。同时,它们的巢也会定期移动。因此,新的寻优算法展现出的特点是:蚁群可以在临近地区被称为搜寻点的地方并行地进行任意的搜寻,所搜寻的位置是一个称之为巢的邻近区域。在固定的时间间隔内,巢会移动。相应地,一个新的并行搜寻过程又将开始。可以把这个称之为 API 的算法运用到某些数字最优化问题求解中,并得到良好的结果。

2.3 蚁群算法与其他智能算法的结合

应用反向传播算法(BP 算法)的神经网络是应用非常广泛的一种多层前馈型神经网络模型,但其算法存在着求解精度低、搜索速度慢、易于陷入局部极小的缺点。蚁群算法可以说是一种新型的模拟进化算法,有正反馈、分布式计算、启发性收敛等特性。这些特性可使得搜索过程加快,从而易于实现分布式寻优功能。将蚁群算法和神经网络相结合,就可实现非线性模型的辨识问题,并有论文中将其用于倒立摆的控制。基于蚁群算法的多层前馈型神经网络仿真实验表明,用蚁群算法来训练神经网络,可兼有神经网

络的全局映射能力和蚁群算法快速全局收敛的性能。

另有人对蚁群中的振荡模式和混沌状态的细胞神经网络建模进行了实例研究。在此项研究中,发现产生同步震荡运动状态的细胞神经网络模型能使一系列混乱的动态元素规则地排列在二维空间的格子中,而其基本的动态系统总是和自己的周围环境发生相互作用。蚁群的行为模式就是建立这种系统的原型。同时,通过分析一些相关的限制条件可以证明,当采用 32 位精度分析时,采用 16 位精度可能产生的伪震荡就会消失。

在这些结合性研究中可以看出,很难说在这些智能算法中有一种胜过所有算法的绝对模型。这从哲学上讲也是不可能的。智能计算是一个运用广泛的研究领域,其中必然有各种值得研究的智能模型,各模型也必然会有其相应的适用领域。当然,蚁群算法必然也会有一席之地,但一定要说蚁群算法在所有领域都具有绝对的优越特征是不合情理的,否则也就不会存在事物的多样性了。

2.4 蚁群算法的仿真和实现

在对蚁群算法的合理应用领域和应用模式还没有定论的时候,在各种典型情况下对其进行仿真研究和实现模式研究是一个合理的选择。

2.4.1 蚁群的行为模型及模拟

首先,对蚁群行为模型的仿真是值得研究的。例如,对各种新的觅食行为类型的研究,可以通过新的量度设计来考察这两者的性能,并通过计算机仿真来衡量其具体作用。

另外,还可以利用许多智能模型来进行蚁群的行为模拟和仿真研究。许多进化算法,如遗传算法(GA)和遗传编码(GP)等,可被用作解决复杂系统建模和最优化的基本工具。一般来说,GA 具有链状的基因,而 GP 具有树状的基因。所有的 GP 适用于构造复杂的项目,它能被应用到很多实际的问题中。但是因为它的基

因内区和肿胀问题,GP 很难被用于解的搜索。于是有人提出了一种新的被称为遗传网络编码(GNP)的进化方法,它的基因具有一种网络状结构,用以克服 GP 较低的搜索效率。这种方法已被应用于蚂蚁的行为模拟问题,用以研究 GNP 的有效性。其性能通过 GNP 和 GP 在模拟蚂蚁行为中得到了比较。

2.4.2 蚁群算法的动态仿真分析

可以说,蚁群算法是一类模拟生物群体突现聚集行为的非经典算法。在蚁群算法的仿真中,首先应描述相对简单的蚂蚁系统及其简单蚁群算法,并对其进行合理的计算机程序模拟与动力系统仿真。有结果表明,简单的蚂蚁系统中存在着规模聚集效应。当蚁群的规模超过某一临界值时,蚂蚁的行为开始向有序的方向收敛,并最终稳定在一种有序状态。

2.4.3 蚁群算法的仿真实现

在运行时间可重置的结构下,对于所执行的迭代式随机宏启发式策略,如何使运行规模压缩,从而使其与不允许运行时间重置的通常结构相比能得到更好的解答质量,这个问题是非常值得研究的。有论文研究了在动态可重构的 Mesh 结构下,如何执行蚁群优化算法。文中讨论了蚁群算法的执行问题,使算法的收敛性能被用于动态降低执行任务所需的 Submesh 尺寸压缩。另外,该文还提出了一种方法来促进蚁群算法得到更快的收敛进程,这样就在运行时间重构的任务上增强了蚁群算法功能。该功能被用于蚁群算法的重复运行,使针对给定问题实例求解的蚁群算法在重复运行过程中能显著提高所获解答的质量。

蚁群智能的问题求解模式是具有在工程问题求解中进行实际应用的可能性的。现在,模拟生物和自然系统来解决复杂优化问题的方法是非常流行的。许多传统的方法不能解决这一类问题,另一方面,许多问题在没有人的帮助下自然界就已经解决了。群体智能模式研究(包含蚁群算法研究)作为一个领域就是提出这

种方法的基础。如前所述，群体智能模型可以实现如同蚂蚁、蜜蜂、白蚁一样在没有直接信息传递的情况下互助合作的集体行为。这种智能的特殊的协作分布式问题求解功能，能被应用到许多分布式问题求解领域，如机器人、网络、经济和环境领域等。因此，通过具体研究，总结出一些群体智能系统的相关准则，以及具体的群体智能技术和新的实现方法是很重要的。

第3章　蚁群算法的应用综述

本章中，作者将从优化问题求解、交通、计算机、机器人研究、电力系统、通信、化工及其他工程应用领域出发，全面综述蚁群算法在其中的应用成果，并作适当评述。

3.1　优化问题求解

3.1.1　组合优化问题求解领域

组合优化问题求解领域是蚁群算法提出以来最成功的应用领域，其代表为旅行商（TSP）问题的求解成果。蚁群算法对 TSP 问题求解具有典型的适用特征，在 TSP 问题的求解中，还可以应用一些相关准则来改进蚁群算法的性能。这些准则是在 20 世纪 60 年代德国和美国科学家研究成果的基础上得到的。其中之一是精英策略，因为这能够模拟真实情况下的蚂蚁，通过信息素所进行的物质交流和信息交流使蚁群协同工作，从而解决单个蚂蚁所无法解决的优化问题。蚂蚁在觅食活动中，在食物与巢穴之间的路径上留下信息素，随着时间的积累，较短路径上的信息素必然相对较浓，而蚂蚁就会倾向于沿信息素较浓的路径往返于巢穴与食物之间。经过一段时间的迭代，就可发现从巢穴到食物之间的较短路径。M. Dorigo 所发表的一篇经典论文系统论述了解决 TSP 问题的这种蚁群协作学习方法。在寻求解答时，蚂蚁的相互合作利用的是存储在 TSP 路径上的信息素这种间接的信息交换形式。实验研究的结果表明，蚁群系统的性能要好于其他诸如模拟退火和进化计算等启发式寻优算法。这是 M. Dorigo 通过实例仿真，比较 ACS-3-opt（一种具有局部搜索特征的蚁群算法版本）和其他一

些解决对称和非对称 TSP 问题的性能最佳算法而得出的结论。另有研究者提出,从更多的方面来模仿真实自然界中蚂蚁的行为,可以更为合理地制定信息素的动态挥发规则;用动态蚁群算法来解决 TSP 问题,多方面来模拟实际的蚂蚁行为,得到了较好的算法性能。

另外,为保证算法的全局稳定性,在基本蚁群算法基础上,还可设计一种新的随机扰动型蚁群算法,并将其应用于求解复杂 TSP 问题。该算法包含两个重要方面:一是可以采用倒指数曲线来描述其扰动因子;二是需要设计相应的随机选择策略和扰动策略。数值模拟表明,该类算法可以有效克服基本蚁群算法计算时间较长和容易出现停滞现象的缺陷,具有更好的全局搜索能力。

在蚁群算法与其他算法的混合模型中,有人研究了求解 TSP 问题的混合型蚁群算法,并以 att532(美国 532 个城市)为例给出了计算实验结果,说明了混合型蚁群算法能改进标准蚁群算法的计算效率和计算结果的质量。另外,根据多目标优化问题的性质,需要提出一种在多个目标间权衡的评价指标,这在蚁群算法解决多目标 TSP 问题的实现过程中也可加以证实。

另外,增强式学习也是一种有效的学习方法,可用来解决学习者事先不了解的环境情况。蚁群系统本身提供了一种在合作者之间间接通信的方法,这已经被证明是一种解决组合优化问题的有效方法。基于蚁群算法(ACS)中的间接通信综合方法和增强式学习(QL)中增强值的迭代策略,有论文提出了 Q-ACS 多路合作学习方法,它能被应用于 Markov 决策过程和组合优化问题求解。Q-ACS 方法的优点是学习者能分享中间过程,这些中间过程有利于使用被积累的知识,同时还可有效地利用已经学习到的新知识。如果更进一步考虑到访问的时间,该论文还提出了一种 T-ACS 多路学习方法。该方法的优点是:在学习过程中,学习者能够获得更好的、有利于探测的策略。同时,考虑到 Q-ACS 和 T-ACS 从总体

上说是同类的多路学习方法，如果考虑到不同的多路非直接媒体通信，该论文还展示了一种不同的 QL 方法，即 D-ACS。它合成了 Q-ACS 和 T-ACS 的学习策略，采用不同的增强知识来进行策略更新。在该论文的方法中，通过给 Agent 一种简单的方法来交换信息，在增加价值时，所有 Agent 的普通模型都被更新。这种方法具备了积极探测未知环境以及有效拓展所学习到的知识这两种优点，能有效地使用组合优化方法来解决问题。关于猎人游戏问题和旅行商（TSP）问题的实验结果，证明了该方法可以和当今各个领域的启发式方法相竞争。

蚁群算法是一种新型的模拟进化算法，但它也存在一些缺点，特别是在规模大的问题求解中，表现出了计算时间较长、容易陷入局部极小等缺点。如果在基本人工蚁群算法的基础上引入遗传算子，同时为保持种群的多样性，在选择策略中引入感觉阈值，另外将分工的思想引入蚁群算法中，可以形成被称为具有分工特征的蚁群算法，并可将其应用到 TSP 问题求解和一些函数优化的实例中。有实验结果表明，这种改进方式是有效的。

另外，针对传统蚁群算法加速收敛与早熟、停滞现象之间的矛盾，也有文章模仿蚂蚁感觉和知觉行为，提出了一种新的蚁群优化算法，使蚂蚁受显意识和潜意识的相互作用来选择路径，同时自适应地修改路径上的信息量。可以通过多种不同规模的对称和不对称旅行商问题为例对此进行仿真。结果表明，该算法具有较好的收敛速度和稳定性，比较适合求解城市数目较多的 TSP 问题。

如果将改进型蚁群算法用于求解旅行 Agent 问题，则有论文提出了一种改进型蚁群算法。其中蚂蚁之间通过外激素进行间接交流而达到合作的目的，并在利用已有信息与探索新解并重的策略指导下给出所求解问题的最优解。另外，由于遗传算子的引入及全局更新规则的修正，使算法不再易于陷入局部极小。论文采用改进型蚁群算法来求解一类复杂的组合优化问题——旅行

Agent问题,取得了满意的效果。实验结果表明,改进型蚁群算法具有鲁棒性强、自适应、并行化、正反馈等优点。

3.1.2 调度问题求解领域

在过去的几十年中,研究人员已经发展了很多适用于调度问题的简单的启发式求解方法,但这些启发式方法的一个主要特点是它们求解的鲁棒性不强。而蚁群算法是一种后启发式算法,对于启发策略而言具有某种改进效果,现在已被较好地应用到调度问题求解中。为了证明蚁群算法的有效性,有论文对加权单机延迟问题进行了计算研究。结果表明,这种方法能够有效地改进各种启发式算法的鲁棒性,并在该基准问题求解上要优于现存的启发式方法。论文证明,在解决问题的质量和计算的代价上,蚁群算法是一种有效解决时序问题的方法。另有论文在单机调度问题上系统比较了蚁群算法和其他启发式算法的计算效果。在单机调度模型的应用上,其设定时间是依赖于序列的,而寻优目标是使总体延迟程度最小化。论文所描述的蚁群算法具有在传输规律上使用预测信息的新特征,这种特征对执行过程有重要的改进。将此算法和遗传算法、模拟退火算法、局部搜索算法和分枝定界法进行比较后,该文作者指出,蚁群算法对于大型问题具有一定的优越性和竞争力。

如果基于全局信息素评价而改进蚁群算法,则其在单机调度中的应用同样能够证明算法的有效性。有论文进行了蚁群算法对于加权和未加权单机延迟问题求解的比较研究,所提算法相对于一般的蚁群算法具有一些新的寻优特征。主要的特征在于其中的智能体不仅能利用本地信息素的相关信息来指导自己的搜索路径,还利用了全局信息素的相关信息。研究还表明,这种算法经优先级顺序的启发式调整,还可以得到进一步的改进。

对Flowshop(流水作业)问题的蚁群优化调度方法的研究,目前也被广大研究者所重视。有论文提出了一种新颖的蚁群优化算

法,在该算法中,流水作业调度问题以结点或弧模式有向图表示,人工蚁受有向图上信息素踪迹的指引,在图上进行搜索,并逐步构造出问题的可行解。算法中的信息素踪迹更新过程作为蚁群间的间接通信机制,能引导整个蚁群收敛到问题的优化解;信息素踪迹更新过程中的停滞状态脱离机制以及信息素踪迹限制机制能帮助人工蚁跳出局部最优解。算法局部搜索过程中采用的基于关键路径的邻域结构可缩小问题的搜索空间。与其他算法在 Taillard 流水作业调度测试问题集上的比较试验表明,该算法性能更优,且具有更强的自适应和鲁棒性。

根据蚁群路径寻优行为模型及其与混流车间调度的相似性,还有论文提出混流车间的蚂蚁调度算法。该算法通过试错来区分加工路线的优劣。实验表明,该算法性能优于启发式算法,可以用来求解随机加工时间的调度问题,并对车间内外部环境变化具有良好的自适应性。

如果将蚁群算法用于解决 $n/m/P/C_{max}$ 问题,这也是一个 NP 难度的调度问题。该问题中需要寻找一种处理顺序,使得 n 种不同的工作在 m 台不同的机器上能够同时运行,从而使生产周期最短。仿真计算包括著名的 Taillard 基准问题。在此问题求解中,有论文中将该算法与其他优化搜索算法如遗传算法、模拟退火算法和邻域搜索法等进行比较,结果证明,蚁群算法对于 $n/m/P/C_{max}$ 问题是一种更为有效的自主启发式算法。

在双机 Flowshop 调度问题和多机调度问题求解中,蚁群算法也能起到很好的作用。双机 Flowshop 调度问题的目标是总的执行时间和总的生产时间最小化,后者的最优化被认为要优先于前者的最优化。这个问题也是一个 NP 难度问题。蚁群算法中还可使用模拟退火算法和局部搜索算法。有计算实验表明,相比于其他的算法,蚁群算法的效率更高。另外还有人比较了算法的总体完成时间。

对于置换型流程调度问题寻优,需要考虑的是最小化构造时间间隔和整个任务的流程时间等。有人提出了两种蚁群优化算法来分析和解决置换流程的调度问题。第一种算法扩展了由 Stutzle 提出的蚁群算法(即最大最小蚁群系统)的思想,结合了由 D. Merkle 和 M. Middendorf 提出的加法规则和一种最近提出的局部搜索技术。第二种蚁群算法被应用于由 E. D. Taillard 提出的 90 个基准问题。首先,作者根据最小构造时间间隔,比较了该论文提出的 MMAS 和双蚁群算法解决方法以及由 E. D. Taillard 提出的启发式解决方法。对照结果表明,一般而言,双蚁群算法性能要优于 MMAS。随后,考虑到最小化整个任务的流程时间这一目标,论文中又比较了所提的第二种解决基准问题的蚁群算法,对照结果表明,文中提出的蚁群算法要明显优于由相关作者分析的启发式方法。在 90 个所研究问题中,有 83 个表明:双蚁群算法比相关作者所提出的解决方法要更好。

在等价并行机械的时序安排中,蚁群优化算法也能得到很好的应用。等价并行机械的时序安排就是最小化生产时间,这对于生产过程的配置问题非常重要。随着问题范围的增大,在解决这个问题的过程中会出现许多困难。而蚁群最优化算法在解决该组合最优问题中显示了很大的优点,它对于实际应用问题具有高效率和适宜性的特征。有相关论文提出了利用蚁群算法在机械时序安排中进行最小化生产时间问题求解的应用。利用两个不同规模的例子可以证明,该算法对于大范围的并行机械的时序问题求解具有有效性和适应性。相对于其他启发式算法如模拟退火法、遗传算法等,该算法更具优势。

在基于蚁群算法的柔性制造系统调度问题求解中,由于在柔性制造系统中一个工作要多个机械一起完成,因此柔性制造系统的时序问题也是一个非常困难的计算问题。有论文在所涉及的蚁群算法框架中,通过对基本算法进行一定的修改,使它更适合于解

决特定的柔性制造系统调度问题。该方法是以节点和曲线描述过程的,基于图形表示方法,描述从过程的一个状态变化到另一个状态。单个的智能体(蚂蚁)从原来的节点通过所有被访问的节点移动到最后的节点,算法的解就是收集所有智能体所找到的解。信息素在所有蚂蚁找到它们各自的解之后得到更新。该算法具有各种较好的寻优特征,如避免停滞、防止过早收敛等。在该论文研究中,将接近最优的解认为是柔性制造系统调度问题的解。该问题也是一个 NP 难度问题,论文所提算法以较少的计算量得到了稳定的优化解,并进一步进行了其他计算实验来验证系统不同的参数对性能所产生的影响。

另有人使用蚁群系统来进行多道旋转操作工序的调度和优化,将切断过程分为粗磨和休整两个阶段,而加工参数由最小单位生产成本和不同的实际加工限制条件决定。结果表明,论文提出的蚁群结构和其他研究者提出的不同算法相比是有效的。

在基于蚁群算法的资源受限工程调度问题求解中,可以利用蚁群算法将两种信息素评估方法相结合,用于寻找新的解答。算法在运行过程中,可以改变启发策略,对蚁群策略施加影响,以避免精英蚂蚁遗忘最佳解答的可能性。有论文利用一组在图书馆工程调度中得到的基准问题对蚁群算法加以测试,并与其他的自适应算法,包括遗传算法、模拟退火算法、禁忌搜索算法和不同采样率方法相比较。结果证明,该算法的平均性能是最佳的。对于几乎三分之一的以前不能得到最优解的基准问题,运用该算法之后都能找到新的最优解。

3.1.3 规划问题求解领域

规划问题实际上是一种受约束的优化问题。蚁群算法是一种新的自适应组合优化算法,它也可被用到实际的规划问题,如空间布局问题的求解中。在此类问题中,需要解决在办公区内具有一定管理功能人员的办公室位置最佳分配问题。这个问题是这样提

出的,例如:一个商业组织想要在它的办公区内减少(或最小化)总的人员移动,以改进其工作效率,则需要研究相关的优化算法。有论文将蚁群算法应用到了空间布局的问题求解中。文中通过禁忌搜索法(TS)的局部改进来提高蚁群算法的相关性能,并且举了两个例子来说明该算法对于大范围空间布局问题的设计是一种可行的最优化方法。

在铸铝过程的单机规划问题求解中,有论文提出了一种带扩展的蚁群算法。基于带多重可视化矩阵的蚁群最优化算法,该工业时序规划问题可被视为是具有序列依赖特征的。求解过程中必须将三个分立目标最小化,多重可视化矩阵的使用可以在多目标优化问题求解中提高解的质量。

3.1.4 约束优化问题求解

进化算法的理论对于数字约束处理问题的求解是适用的。另外,在研究约束满足问题时,另有基于蚁群算法的求解方法。有论文通过全面考察利用进化算法的流行的约束处理技术,运用从单变量的惩罚函数到其他复杂函数的手段来对相关进化算法加以评论。这些复杂的算法模拟了生物上的免疫系统,或者就是由蚁群行为而激发的。文中简要地描述了各种算法,还提出了这些算法的优点和缺点,并且将提出的 Dominance-based(以优势为基础)的算法和一些 Penalty-based(以惩罚为基础)的方法进行了比较,最后提出一些相对于特定的方法怎样选择适合的约束处理技术的观点。

在基于蚁群算法的约束问题求解方法中,可利用人工蚂蚁智能体,通过寻找信息素的踪迹来追踪在搜索范围内最可能的解答范围。这些信息素是用来引导搜索方向的,就像启发式是用来选择指定变量的赋值那样。有论文介绍了用来解决约束问题求解中基本的蚁群算法是怎样和局部搜索技术相结合,从而得到改进的。其中首先需要一个预处理阶段,目标是花较少的成本进行最大范围的搜索。在求解过程中,该文作者较快地得到了一个比较满意

的寻优结果，并且评估了这种方法对于随机二进制问题的求解特征。

3.1.5 连续空间优化问题求解

蚁群算法对于最优化问题，特别是顺序类型的组合最优化问题是一个新颖的计算方法。而有论文提出了适应性蚁群算法，用来解决连续空间的最优化问题。该算法使用原函数为基础的启发式信息去更新原来的信息素，并且以此来过滤后备解。通过智能体参照目标量来进行搜索路径的自调整，可以很快地达到全局最优解。比较适应性蚁群算法和基本的蚁群算法以及最小二乘法在处理两个具有大量极端值的基准问题上的结果显示，这种改进算法具有更强的有效性和可靠性。

在基于蚁群算法的连续函数优化问题求解中，蚁群最优化算法相比于其他随机搜索算法具有更大的可行性。有论文将该算法用于各种不同的基准测试函数，在受约束的和不受约束的 NLP、MLP 和 MINLP 最优化问题中进行了验证。结果证明，这种新颖的算法适用于各种类型的连续函数，也能用于大范围的过程最优化问题。

针对蚁群算法不太适合于求解连续性优化问题的缺陷，另有论文中所提出的用蚁群算法求解连续空间优化问题的一种方法是，将解空间划分成若干子域，并在蚁群算法的每一次迭代中，首先根据信息量求出解所在的子域，然后在该子域内已有的解中确定解的具体值。文中以非线性规划问题为例，所进行的计算结果表明，该方法比使用模拟退火算法、遗传算法等具有更快的收敛速度。

在单目标和多目标可靠性最优化问题的蚁群算法求解模式中，有论文描述了蚁群优化算法在实际工程中用于解决连续函数和组合优化问题的方法。文中将蚁群算法与增强型 Pareto 适应度配置进程相结合用于解决多目标问题，进而将聚类程序用于精简 Pareto 集合，以维持其多样性，并通过典型范例研究显示了蚁群算

法对解决此类问题的优越性。

在另有一种求解连续空间优化问题的蚁群算法中，主要包括了全局搜索、局部搜索和信息素强度更新规则。在全局搜索过程中，利用信息素强度和启发式函数来确定蚂蚁移动方向。在局部搜索过程中，嵌入了确定性搜索，以改善寻优性能，加快收敛速率。通过一个实例问题的求解，该算法的有效性被加以证实。

在另一类求解连续空间优化问题的自适应蚁群系统算法中，则采用了新的基于目标函数值的启发式信息素分配算法，以及搜索过程中最优解的筛选方法，并根据目标函数来自适应调整蚂蚁的路径搜索行为，从而保证算法快速找到全局最优解。通过一个具有多极值点的连续优化问题求解实例，该方法的有效性得到了证明。

另外，在多选择背包问题的优化求解中，也有蚁群算法的应用范例。该方法充分利用了蚁群算法所具有的正反馈特性，再结合变异参数，使算法既有较快的求解速度，又有较高的求解精度。实验结果表明，采用此算法能快速有效地解决背包问题。

3.1.6 二次指派问题求解

在基于蚁群算法的二次指派问题（或称为二次配置问题）求解中，也有了许多研究成果。有人研究了一种用于二次指派问题的混合蚁群算法，其中蚁群智能结合局部搜索被运用到二次指派问题求解中。该算法使用信息素的信息去修改二次指派问题的结果，而不像传统的蚁群系统那样使用信息素的信息去指导所有的结果。文中对该方法进行了分析，并和一些二次指派问题的启发式算法相比较，如鲁棒和反应式禁忌搜索法，以及混合的遗传算法和模拟退火算法等。实验结果表明，蚁群算法以及混合遗传算法，对于实际的不规则和结构化问题能找到最优解。然而文中也指出了，蚁群算法在处理随机规则和非结构化问题时没有竞争力。

二次指派问题也可用并行的蚁群算法来求解。在该类算法

中,人工蚂蚁之间的合作是由一个信息素矩阵提供的,它扮演了全局存储记忆的角色。在搜索空间的探索工作由信息素水平的演化量所引导,而其蒸发模式由局部搜索启发式策略来加以限制。在多元化阶段的设计中,可引入基于频率矩阵的异化相位设计。文中给出了相关的研究结果,并且证明,所获结果能和其他二次指派问题的寻优算法进行比较。

3.1.7 着色问题求解

着色问题求解实际上也是一种配置类问题。有论文提出用基于蚁群算法的进化搜索策略来解决该配置类问题。该算法通过迭代直到在两个准则下建立一个可行的问题解。这两个准则是信息素因素和满意度因素。使用这样的搜索规律可用着色问题求解来加以说明。当与此类问题求解中现存的方法相比较,可以发现,利用这种方法得到的结果还是比较满意的。

3.2 基于蚁群算法的交通过程建模及规划问题求解

由于蚁群算法与实际的交通问题求解有着很强的直接对应特性,故将其直接用于交通过程建模、规划和优化问题求解是非常合理的。

3.2.1 交通过程建模

在现实生活中,有很多突发的交通现象是不能用数学模型来准确地进行分析和解释的,分析这类现象只能通过模拟每个智能体行为的方式来进行。基于群体智能的多智能体系统交通传输模拟,如 Agent 模型,就属于此类方法。它将各个分散的、独立的智能体结合成一个系统,这些智能体可以通过局部知识来影响其他的智能体。这种智能体方法能从下往上地对那些模拟社会性昆虫的特殊智能体进行建模。社会性昆虫(包括蜜蜂、黄蜂、蚂蚁和白蚁等)在地球上生活了数百万年。它们的自然行为主要受其自

治、分布式的机能和自组织能力的影响。社会性昆虫种群告诉我们,系统的任何一个简单的个体通过动态的相互作用就可以完成非常复杂的任务。另一方面,大量的传统工程模型和算法都是以集中化控制为模型的。作者试图借助蚁群算法等自然化的群体智能法则来开发一个旨在解决复杂的交通传输问题的方法。

3.2.2 交通过程优化及导航

通过改进蚁群算法的应用,可以解决 TSP 问题和其他类似优化问题。同样,在水路运输中也可加以类似的应用。有文章分析了上海的内河航道和集装箱的运输问题。集装箱的集散是上海深水港建设的一个重要方面。为了对上海内河航道提出合理的布局,文中推广了蚁群算法,通过随机分布技术对蚁群算法进行了改进;并且通过改进蚁群算法的实施,使上海内河航道更适于集装箱的运输,由此对于上海内河航道的建设提出了合理性的意见。

另有论文展示了一种基于蚁群算法的进化计算模式,用于车辆导航的进化式视觉传感。这种方法能很好地处理数字图像,而且能发现在南极洲的冰雪面上前行的交通工具所留下的踪迹。所提蚁群能进行互相影响和合作,以此来决定所定义到食物地点的最短路径。这种由蚁群激发的策略是一种协作的智能体行为,这些行为者沿着图像像素移动,并且反复地改进初始的较粗糙的解决方案。恶劣的南极圈环境使图像分析问题极具挑战性,从灯光反射,突然改变的光亮情况,到不同的地带倾斜都必须考虑周到。有论文比较了基于蚁群算法的方法和一种较传统的 Hough 方法,并且给出了相关结果。

3.2.3 车辆路线规划问题求解

基于蚁群算法,可以很好地对车辆路线规划问题进行求解,同时可依照计算效果来改进其性能。首先可以将问题进行分解,在此基础上,只要解决各个小的子问题就可以了。通过计算研究和统计学分析,以标准的基准问题和大范围的车辆路线规划问题为

例,表明蚁群算法不仅能改进效率,而且能成为解决真实世界中的实际车辆路线规划问题的工具。

另外,还可将基于信息素传递的互助蚁群搜索方法用于车辆路线规划问题求解。在此类算法中,多智能体可以将问题进行有机分割,然后通过信息素来传递独立的搜索解。这是一种用来模拟真实蚂蚁之间通信模式的方法。通过计算机仿真实验,可以证明这种基于多智能体的互助搜索算法的有效性。

3.3 计算机领域

3.3.1 专家系统问题求解

在传统的计算机人工智能研究领域中,专家系统是其典型模型。其中,有一种基于或然论的专家系统叫做贝叶斯网络。蚁群算法可以在贝叶斯网络中搜寻最佳的消除顺序。大多数利用或然论专家系统的工程工具不是直接由网络推论出来的,而是建立在第二个层面上,这被称为连接树。推论算法的效率取决于连接树的大小,而这个大小又取决于从贝叶斯网络到连接树转变过程中的消除顺序。搜索最佳的消除顺序是一个 NP 难度问题,而且这又可以受到类似算法的启示(如启发式、遗传算法、模拟退火算法等)。有人研究了用于这种搜索算法的一种新的组合优化方法——蚁群算法,通过在复杂网络中的使用,结果发现这种方法是有效的。

另外,基于蚁群优化算法,还可进行学习型贝叶斯网络的构建。对于通过数据学习构建贝叶斯网络的问题,一种重要的方法就是在数据基础上,对于给定的候选网络,用十进制来估计其适应值,然后提供一种搜寻方法去研究这个网络结构。常用的搜索方法是贪婪性爬山算法,或是确定性的爬山算法,或是不确定性的爬山算法。有论文提出,对于学习型贝叶斯网络,蚁群算法是一种新的求解模式,它对于解决其他各种组合优化问题也是非常有效的。文中在三个不同的领域内展开了实验比较工作。

3.3.2 计算机图形学领域

基于蚁群算法,可以设计多边形模拟飞机曲线的最优化方法。有文章介绍了一种利用蚁群搜索算法的新型多边形模拟法。当问题用一个指导性的图标来描述时,问题的目标就变成用图标在问题限制下如何找到最短的封闭线路。在图标中分布了大量的人工智能体,它们之间互相传递路径信息,形成长期的记忆来指导以后的图标搜索。文章研究了蚁群算法在其中的重要特性,该方法与遗传算法、禁忌搜索算法相比,前景是非常诱人的。

另外,在基于 Trimmed NURBS 曲面几何特征的数字化自适应采样中,蚁群算法也可加以应用。在曲面轮廓度误差评价和多边域逼近中,均要求对理论曲面进行数字化自适应采样。在对当前曲面数字化方法分析的基础上,可针对 Trimmed NURBS 曲面,研究在给定精度下的曲面数字化自适应采样方法,从而给出基于曲率特征函数的曲线自适应采样和 NURBS 曲面多边域离散算法。有论文提出了一种对大规模曲面网格点检测路径优化问题很有效的蚁群算法,并进行了计算机仿真实验。结果表明,所提出的规划方法可显著提高数字化采样效率,具有很好的实用价值。

3.3.3 数据挖掘和数据高层综合问题研究

在利用蚁群智能的数据挖掘工作中,有论文介绍了一种数据挖掘的方法,叫做蚂蚁矿工法(基于蚁群的数据挖掘),其目标是从数据中分离出合理的分类规则。这种算法是通过研究真正的蚁群行为和数据挖掘概念而产生出来的。在六个公共域数据集合中,可以比较蚂蚁矿工法和 CN2 法,即另外一种用于分类的数据挖掘算法的性能。结果显示:蚂蚁矿工法比 CN2 法在预测精度上更准确,并且蚂蚁矿工法发现的规律要比 CN2 法发现的简单。

在基于数据的高层综合问题研究中,有一种新的算法,叫做动态蚁群算法。它结合了蚁群优化技术和动态小生境共享策略。这种算法的引人之处在于,它能通过增加启发式权重来加快算法。

这样，不仅使该算法执行简便，而且能与现存的算法很好地进行匹配。在这种增强型算法中，运用了问题的状态结构，且可以通过利用信息素来避免存储容量的爆炸。在这种用于数据通路设计的高层综合算法研究中，可通过一些标准的设计步骤和多项设计技术来完成此项目标。

3.3.4 计算机网络管理和移动计算

在计算机网络管理，尤其是路由算法设计中，蚁群算法有着不可替代的优势。基于移动智能体算法的网络和路由管理是一个很有意义的研究领域。其中，基于代理技术的蚂蚁路由就是一种解决网络路由问题的方法。已经有文献证明了蚁群算法的有效性，并且对蚁群算法的群体成长性质和突变行为进行了初步分析。

为了克服传统蚁群算法收敛速度相对较慢，并容易限于局部最小的缺陷，在某论文所研究的一种基于蚁群算法的多媒体网络多播路由模型中，对蚁群算法进行了改进，在每次循环结束时，保留最优解，自适应地改变挥发度系数。文中引入遗传算法的交叉算子，提出了一种基于蚁群算法的有时延约束的多播路由算法模型。仿真结果表明，基于改进蚁群算法的多播路由算法模型可以稳定地获得优于现有启发式算法的解，是一种有效的多播路由算法，并且该算法也适用于并行执行和应用场合拓展。

在一种基于拓展性蚂蚁路径寻优算法的网络路由管理模式中，有人研究了一种以活动智能体搜索作为网络路由选择的新方法，叫做蚁群路径寻优方法。在蚁群路径算法中，从智能体种群的成长和跳跃行为中得到了一些简单的结果，所关注的焦点是将所期望的智能体集中到节点上去。特别是，在带宽受到限制的网络中，大量的智能体在每个网络中的繁殖是非常重要的。有论文提出了两种拓展的蚁群路径算法，然后基于算法中智能体的繁殖行为，在网络的节点和通道方面进行了进一步的分析。

在基于分布式蚁群算法的网络有效巡逻问题研究中，也可考

虑用一组简单的无记忆的机械智能体来探索整个网络。整个网络是没有指导性地图的,其目标不是要完整地探索整个图,而是要找到最接近一致的穿越地图边缘的频率。有论文提出和分析了一种简单的多智能体探索算法——蚁群算法。该方法表明,该简单的智能体经过过渡过程,能进入拓展欧拉循环的周期性运动。在这个过程中,穿越过程可以达到一个固定的次数。论文还可以证明,如果网络是欧拉型的,单个智能体在 $2|E|D$ 次数内就能完成一个欧拉循环。其中,$|E|$ 表示地图的边缘数,D 表示直径。作为 k 个智能体,在两个因素的影响下,经过 $2(1+1/k)|E|D$ 步,所访问网络边缘的数目将达到平衡。算法的另外各个方面的特性也可通过仿真加以证明。

3.4 机器人设计及控制

3.4.1 机器人设计

蚁群的信息素能否作为机器人设计中仿效的一个例子呢?答案为能!大量自然生物对于化学信号都极为敏感,这对于其生存是非常重要的,然而一般系统中很难将这种敏感模型移植到机器人系统中。而昆虫为了有效地适应变化的和无组织的真实自然环境,则采用了多种策略。产生和探测气味是这些策略中最基本的。我们这里可以采用相似的技术去改进机器人系统的执行能力。有人已经设计了一种方法,将蚂蚁的跟踪信息素的能力移植到机器人系统中。在此方法中,一个机器蚂蚁将装配一对气味敏感的触角,系统将装入可以像黑蚂蚁那样跟踪路线的程序。这种机器人的控制策略和跟踪信息素的实验已经完成。

另外,还可以将蚁群的合作性思想用于机器人的设计过程中。在基于兵蚁合作性设计的起重机器人论文中,介绍了所设计的一种统一的小机器人,并通过它们的相互作用来展示其用途。这类新的简单可互换的机器人叫做兵蚁机器人,它们以一个团队来完

成各种任务,同时作出各种相互协作的决定。所有的机械和电路设计都取决于团队的群体动态行为。这种兵蚁机器人的优点在于相对于普通机器人来说的模块化,它们能在短时间里很快完成协作重组;如果其中一个坏了,不会破坏整个团队剩下的功能,替换它也不会花费昂贵。

另外,科学家和工程师们还希望能受到自然界的启发来解决机器人控制设计问题上的难点。对于在机器人控制设计中的自然法则,要解决的问题主要有:腿的自我稳定,创造出可行的跳跃动作,设计算法来完成感觉行为,以及通过生物灵感来构成机器人行为。通过从螃蟹、蟑螂、蚂蚁、蜜蜂等社会性生物所得到的启示,我们可以预知,这种机器人至少在理论上是可行的。

3.4.2 机器人控制及协调

使用蒸发信息素的策略,可以完成蚂蚁机器人的分布式协调控制。蚂蚁和其他昆虫都知道如何利用信息素来进行各种信息的传递,并且完成各类互助任务。由此人们设想,是否能够研究具有这种能力的一类机器人,它们通过最近的信息素来传递信息,在没有事先绘制地图的建筑里完成清理地面的任务,或者横穿一种并不清楚的地域;是否可以研究能够散发随时间蒸发的信息素的机器人,该机器人能在任何它们到达的地点对所检测到的气味强度进行评价。在此项已经报道的研究中,所抽象出的模型是一个分布式的多智能体系统,其中的各个蚂蚁机器人共享着存储器,并能够沿着地板上的路线进行移动。其中已有三种分布式情况下的行进方法。随着时间的推进和气味的逐渐消失,这三种方法最后都能完成任务,其中两种是在多项式时间内完成的。和现存的搜索方法,如深度搜索法不同,所研究的算法具有一定适应性。在全部的协调控制过程中,尽管一些智能体会死亡或者地图会改变,但只要地图路线保持连续,所设计的机器人系统就能保证完成任务。系统的另一个相互作用过程的优点就是,只要增加运行时间,各智

能体就可以使用带有噪声的信息。

在基于蚁群算法的机器人协作运输问题的研究中，所依据的生物原理是：在许多种类的蚂蚁中，工蚁会通过相互协作去寻找食物。通常一只蚂蚁发现食物后，先试图自己去搬动，当几次失败后，就通过直接接触或者放出化学信号来寻找大量的公蚁。当大队的蚂蚁搬动食物时，蚂蚁会不断改变位置和队列，直到把食物搬回巢穴为止。这里，可以用机器人来模拟完成这种工作。虽然机器人系统不能表现出很高的效率，但是通过一组机器人来解决分布式问题，这还是一个很好的例子。这就是蚂蚁合作运输的模型实现。

3.4.3 移动式机器人导航

在应用昆虫导航策略的移动式机器人装置研究中，我们可以知道，在复杂环境中的导航策略对于动物和机器人来说都是非常重要的。许多动物使用组合各种不同的策略来得到它们在环境中重要的位置信息。例如，沙漠蚂蚁能够在沙漠中为找寻食物而走出数百米，然后几乎是沿着直线准确回到自己的巢穴中。它们能做到这些，多亏了三种主要策略，即路径综合、视觉导航和系统搜索。在模拟沙漠蚂蚁导航策略的研究中，可以应用综合方法论去得到额外的导航信息，就像沙漠蚂蚁的导航行为那样。受昆虫导航系统的启发，有研究者成功发明了具有路径综合及视觉导航机制的系统，这些系统被成功安装在移动式机器人 Sahabot 2 身上。一方面，从实验得到的结果验证了研究者考察相关生物模型所得到的模型；另一方面，有昆虫简单的导航策略作为指导，从实验中可以得到移动式机器人计算简单的导航方法。

3.5 电力系统应用

3.5.1 电力系统优化

对于电力系统优化，主要涉及到如下几个方面：

(1) 确定在规划期内何时、何地,建立何种类型、多大容量的一批发电厂,从而最经济合理地满足电力负荷发展的需要。

(2) 在已知规划水平年的预测负荷和电源规划限制的基础上,根据现有网络和给定参数,合理布局新建线路,使电网设计能适应负荷要求、灵活可靠,并且满足安全运行的要求,同时满足其经济性的要求。

(3) 进行无功优化,使电力系统在满足无功负荷需求和电压水平要求的前提下,充分发挥现有的各种调压措施的作用,寻求合理的无功补偿点和最佳补偿容量,使所耗费用最低。

(4) 在电力工业中,如果要引入竞争机制,就需要发展电力市场,由用户选择发电公司,根据用户对每一个发电公司的购电报价和每一个发电公司的保留电价,在满足各种约束的条件下,使报价合理的用户可以从电网中的任意发电公司购得所需数量的电能。

上述许多问题存在着维数灾难、局部最优和约束条件及目标函数不易处理等特点,随着系统规模的增大,待选方案将显著增多,发生"组合爆炸"现象,有的还属于典型的非凸多峰问题。除了全局最优解外,一般还存在若干局部最优解。因此用传统的优化方法解决这类问题,往往得不到满意的结果,在系统规模较大时更是如此。

本书所介绍的蚁群优化算法正适用于组合优化问题求解,在电力系统中已初步显示出其可行性。首先,在继电控制系统中,元件的连接关系可以使用网络拓扑图描述,元件间的接线路径优化类似于 TSP 问题,这属于 NP 完全的组合优化问题。在基于蚁群并行算法的电气接线路径优化及仿真中,已有人建立了适用于继电系统接线路径优化的蚁群算法模型,并在消息传递界面的基础上实现了算法的并行化,还通过对算法初始参数进行的仿真分析,确定了各参数的最佳取值范围。实验结果证明,在参数选择适当的情况下,蚁群算法具有很好的全局搜索能力和较快的收敛速度。

另外,有人对一种求解最优机组组合问题的随机扰动蚁群优化算法进行了深入研究;并且针对蚁群优化算法中易出现的停滞现象,设计出了一种新颖的随机扰动蚁群优化算法。该算法包含了两个重要方面:一是提出了采用倒指数曲线来描述的扰动因子;二是设计出了相应的随机选择策略和扰动策略。此外,相关论文还对该算法中参数的选取方法及取值范围进行了研究和探讨。在用该算法求解最优机组组合问题过程中,在模型的转化、约束项的处理等方面进行了深入的分析。通过对两个测试系统进行计算,并与基本蚁群算法的计算效果进行比较,证明了该算法可以有效地克服基本蚁群算法计算时间较长和容易出现停滞现象的缺陷,具有更好的全局优化能力。

3.5.2 电力系统负荷分配及调度

在此领域的研究中,已经有文章提出了一种可用于求解一般非凸、非线性约束优化问题的广义蚁群算法,以用于求解复杂的非凸、非线性电力系统的经济负荷分配问题。与用于组合优化问题的蚁群算法类似,所提算法具有正反馈、分布式计算和贪婪式启发搜索特征。文中基于不动点理论,给出了该算法收敛的充分条件。经过多个算例结果表明,文中所提出的算法是有效可行的。

在解决电力系统中的热电经济分配问题时,其难度在于它受到多目标的限制。相互作用的热电组合使得一个可行的作用域都很难找到,更不用说最佳方案了。在此,蚁群搜索算法可用于此类问题,其搜索能力非常诱人。当然,也存在一些困难,如处理各种限制条件和过早收敛等问题。有人提出将蚁群算法和其他搜索方法相结合,以改进它的性能,在此领域中的应用结果还是令人满意的。

3.5.3 发电规划及调度问题求解

利用蚁群算法,可以加强水力发电的时序安排。现已有人提出基于蚁群算法来解决水力发电的时序安排问题。利用这种方法

解决问题，首先要确定多级序列的搜索空间，然后，通过收集相互协作的智能体蚂蚁，就能很快得到对于序列问题的近似最优解。在该算法中，用到了状态迁移规则、局部信息更新规则和全局信息更新规则，使计算显得比较容易。因为这种方法能使智能体同时运行，而且能很好地防止过程中断，所以它的最优化能力得到了加强。该方法已经通过实用数据在 Taipower 系统中得到了测试。测试结果证明，这种方法是可行且有效的。

在基于蚁群算法的具有随机扰动的机组组合问题研究中，已有人提出了一种新的具有随机扰动特征的蚁群优化算法。这种算法包括两个方面：一方面，扰动用反指数函数进行表示；另一方面，设计相应的转移策略，随机选择扰动行为。该算法首先被用来解决机组组合问题，同时考虑了一些相应的技术问题，如相同和不相同限制条件、状态的迁移等等。数字仿真的结果证明，这种新的方法具有较强的全局寻优能力，在减少 CPU 运行时间和防止搜索停滞方面有很好的效果。

另外，基于蚁群算法还可进行电力系统最优切换方法的研究。有论文提出了一个合作的 Agent 算法，即用于最优开关布置的蚁群系统算法。开关布置对于电力系统自动化来说是一个有用的工具，因为它能减少切换成本而不需要附加的资金投资。开关布置的确切描述式是一个带有非线性约束的组合优化问题，该项研究的一个主要目标是调查在电力系统优化问题中基于蚁群算法的应用效果。测试结果表明，基于蚁群算法能提供一个比较适宜的开关布置新模式。在测试结果中，作者比较了这种方法和一种基于遗传算法的布置方法，以论证这种方法的价值。

在发电调度问题求解中，还有人使用蚁群智能搜索方法进行短期发电调度问题的时序求解。这是一种通用的互助智能体方法，所提到的基于蚁群算法的搜索策略，能解决短期的热电系统调度问题。这里需要研究的是选择性的智能搜索方法在电力系统优

化中的应用模式，特别是在短期发电调度问题中的应用。一组互助的智能体（蚂蚁），能相互合作地寻找短期热电调度问题的最优解。在所提算法中，状态迁移规则、全局和局部更新规则同样被用以确保得到最优解。一旦所有的蚂蚁都完成了自己的游历，就运用全局信息的更新规则，如此重复，直到满足条件为止。这个有效的方法已经解决了电力系统模型中的发电调度问题。

3.5.4 电力系统故障分析

对于电力系统故障地点的估计，也可采用一种新的蚁群算法。由于估计的信息来自保护继电器和电路断路器，所有故障部分的估计都可被看作是一个组合最优化问题。蚁群算法通过智能体可以进行并行处理，并且在过程中没有任何监督，因此这种方法是适合处理此类组合最优化问题的。有电力系统的实例分析可以验证蚁群算法的有效性。通过这类测试可以发现，蚁群算法对于故障地点估计类问题的求解也有较好的应用前景。

3.6 通信领域

通信领域有很多值得研究的优化类问题，包括通信网络的负载平衡问题、路由选择问题、频率分配问题和相关的系统综合问题等。另外，在移动通信系统中，移动计算也是一个很值得研究的领域。许多类似问题都可转化为组合优化类问题求解，因此在这些领域中蚁群算法也有很好的应用前景。

3.6.1 路由选择及负载、资源配置问题

在进行具有或然性的路由安排蚁群算法研究时，人们发现在单个蚂蚁行程中，并不是以最大概率的信息素作为标准，而是随机地以这些信息素为标准。这个准则对于最小化节点拥塞问题求解是非常有用的。另外具有附加反信息素机制的人造智能体（蚁群智能体）也能更好地平衡网络。在适当条件下的路由安排策略是值得研究的。现已有人在进行基于蚁群算法的具有信息素和反信

息素的或然性路由安排机制的研究。

在 LEO 通信卫星广播网中,蚁群优化算法被认为是其中自适应路由设计的一种很有前景的算法工具。有人用蚁群算法模拟了具有 72 个 LEO 节点和 121 个地球站点的卫星通信网。为了评估算法不同部件的相对重要性,采用了三种不同的蚁群算法变量。所得结论是:在不同通信量条件和忽略路由带宽的情况下,所提到的具有优化变量的蚁群算法性能要优于标准连接状态下的算法性能。

在相关路由算法研究中,另有一种智能路由算法,称为 Q 代理,其行动仅基于代理环境的交互作用。此算法可结合三种学习策略进行(Q-Learning、双重增强式学习和基于蚁群行为的学习),并结合两种更深入的机制来改进其适应性。因此,这样提出的算法是由一系列的智能体所组成的,能够独立地且并发地通过网络来搜寻最佳路由。

对于频率分配问题,可以用启发式蚁群系统来解决如何为基站和移动用户之间的无线波分配频率,以使给定区域总的冲突量减到最少。这个问题是具有 NP 难度的,有关它的最优化技术很少有报道。基于蚁群的宏启发式算法可以应用到这个问题的求解之中。这是一种跟踪蚁群最优化范例的方法,可以通过一些标准的问题实例计算结果来证实该方法的有效性。

另外,基于自适应蚁群算法,还可进行多受限网络 QoS 路由的优化问题求解。高速多媒体网络中的路由问题是一个有 QoS 约束的路由问题,该多受限的路由问题也是一个 NP 完全问题。可以采用自适应蚁群算法,建立基于目标函数值的信息素分配策略,并根据目标函数值自适应调整蚂蚁的搜索行为,从而保证搜索的快速有效性,使多受限 QoS 路由优化问题得到很好的解决。

3.6.2 移动计算

在当今通信领域的移动计算中,定位管理是一个非常重要和复杂的问题,因此有必要研究一种算法,使该算法能很容易地针对

这种复杂性,解决定位管理中的各种问题。人工生命技术是近来用以解决这一系列复杂问题的方法。这些技术的解决问题能力来源于它们处理大规模搜索空间的能力,此类大规模搜索空间可出现在许多组合优化问题中。有论文比较了几种著名的人工生命技术对解决定位管理问题的匹配能力。由于这些算法的灵活性和鲁棒性不同,文中采用遗传算法、禁忌搜索和蚁群算法来解决移动计算中的定位问题。在手机的定位管理程序设计中,一些手机被设计成报告手机,移动终端一进入报告手机区域就更新它们的位置(位置更新),这需要用遗传算法、禁忌搜索和其他几种不同的蚁群算法来产生一个规划器。文中通过一系列的测试问题,说明了每种算法的效力,并进行了移动计算中三种人工生命技术的比较。

3.6.3 天线阵列控制

通过使用改进的移动蚁群优化算法,可以控制基本位置的线性天线阵列模式零位。在这种基于改良的移动蚁群优化算法的搜索技术中,只要控制线性天线排列元素的位置,就能进行模式零位的控制。其代价函数中引入了一组权重因子,所设计的功能可以用来改进移动蚁群的优化运算。这里,禁忌搜索的频率存储特征被用于改进移动蚁群优化算法。为了证实技术的有效性,可以引用 Chebyshev 模式中的说明性例子。

另外,可以利用改进的旋转蚁群最优化算法来进行线性天线阵列的无效操控。在带有规定零信号的线性天线阵列模式合成中,蚁群算法是一种新的灵活算法,可以通过仅仅控制每个阵列元素的幅度来达到模式的调控。为了说明该方法的多功能性,在蚁群优化算法所构造的代价函数中,可以引入一系列的权重因子,从而考虑一些相关的设计规范,如旁瓣水平、零位深度以及动态范围比等。

3.6.4 智能网络控制

在智能网络流量实时控制中,存在着 FIPA-compliant 智能体

的概念。在自适应性可测量的柔性通信系统中,其概念都可归结为智能体和多智能体系统。这些系统可及时地提供对不断增加复杂性的交通流量控制和通信网络中的资源管理处理方法。如果基于智能体的控制方法对于网络管理问题求解是成功的,那么对于不同种类的智能体之间的相互影响进行研究,进而形成可以接收的标准也是非常重要的。尽管这些标准在发展,但是如果不能通过特殊的变化去适应通信领域的要求,这就是非常重要的失误。在实时系统中的智能体系统,是不能被看作有足够地址的。这里有两种在智能网络中用于交通流量控制和资源配置问题的智能体系统。一种是基于市场化控制概念的,利用基于市场化的准则的例子,我们发现可以通过加强现存的 FIPA 的规则来满足实时控制的要求;另一种就是基于蚁群优化算法概念的多智能体资源管理系统模式。

在网络业务控制研究中,还可用一种新的神经网络方法来建立有效的网络业务控制器,即利用蚁群算法来训练神经网络的权值,并将其应用到 ATM 的业务控制中,此思路的有效性已经由相关的仿真研究所证实。

另外,在用于两端线网布线的蚁群系统方法中,对于给定的布线平面,首先可根据障碍情况构造包含最短路径信息的强连接图,建立初始气味矩阵,然后使用蚁群算法来搜寻目标路径,直到求出优化解。

3.6.5 相关模块设计和系统综合

这里以人造卫星模块的优化布置设计为例。人造卫星模块的布置设计中,需要关注的是相关结构、性能、服务时间和装配及维护成本。这又是一个重要的 NP 难度问题。其中主要的困难包括工程实践中的数学问题和考虑系统复杂性的综合优化,而且相关的研究很不充分。有论文中提出了一些人造卫星模块优化布置设计的基本策略,其中包括两个阶段:整体(松散)布置设计和细节

布置设计。有论文以国际商业通信卫星的布置设计为例,给出了相应的优化算法。这些算法是整体布置设计所用到的向心平衡法,以及优化布置时所用到的基于蚁群优化算法的准 TSP 问题模型。这些应用示例显示了所提方法的可行性。

另有论文介绍了一种蚁群最优化算法,称之为树蚁(Ant-on-a-tree)算法。这种算法能综合许多算法共同解决一个问题,算法的有效性通过解决一个高层综合问题得到了证明。该高层综合问题中包含了许多设计步骤和算法,用于分别解决每个问题。而树蚁模式能很容易地综合这些算法来限制搜索空间,同时用它们作为启发性权重来指导搜索过程。在搜索过程中,树蚁算法能产生一棵动态的决策树,并有一个类似于分枝和跳跃运算法的实现技术可用于指导决策树的搜索。其中,由蚁群产生的信息素避免了存储量爆炸问题,这是该搜索算法所固有的性质。

3.7 化工领域

蚁群算法可以实现化学过程的动态优化。有文献通过具有六个重要基准的动态优化模型来解释蚁群算法。这种新的计算方法对于处理那些在简单结构中有限制条件的问题非常方便,它只需要很少的栅格点和较短的计算过程就可以达到全局最优。可用具有很大复杂度的化工过程模型来证明,这种优化算法对于解决在化学工程中的大范围寻优过程问题是十分适用的。

对于化工过程这一批量对象的最优设计和规划,蚁群算法这一协作搜索方法已经在其中得到了应用。首先,该算法能够解决多产品批量下行程安排问题的组合最优化;其次,它能够解决在单个产品活动和基本的限制条件下,多产品批量对象设计的连续函数优化问题。蚁群算法在其中的简单应用,以及实例研究的结果,均表明,它能提供快速和正确的解决方法。

另外,一种新的进化方法——蚁群觅食机制(AFM)对于解决

多分支的化学过程最优化设计也是非常有效的。为了证明这一点,有人结合计算结果举了四个例子。将这些结果与数学规划(MP)、禁忌搜索(TS)、遗传算法(GA)和模拟退火算法(SA)的结果相比较,还是满意的。

另外,在分析化学计量学研究中,每一种新算法的诞生都会带动新一轮研究热潮的掀起,并极大地推动化学计量学的发展。然而,在化学计量学中,有关蚁群算法的研究报道还很少。为此,有人首次提出了化学蚁群算法,并将它应用于信息量丰富的导数荧光光谱的解析中,获得了满意的结果。

3.8 工程应用领域

3.8.1 制造过程控制及优化

蚁群算法可以求解相关的流水线平衡问题。该算法能够很好地解决具有并行工作站、随机任务间隔和混合模型这些复杂因素的流水线平衡问题。同样,这种方法也受到社会性昆虫行为的启发,试图将任务分配给各个智能体的。所设计算法中,所有的策略性能指标都得到了优化。相关文献指出,这种方法被用来解决多流水线的平衡问题。而流水线的布局问题求解则可通过各种解决方案来模拟生产过程,以此得到输出的性能测度(如循环时间特性等)。和其他算法,如模拟退火算法等进行比较的结果显示,蚁群算法得到的性能指标更具有竞争力。

3.8.2 工程设计问题求解

在电路生产设计中,可以通过考虑减少放置时间来减少印刷电路版的整体装配时间。在大批量生产的印刷电路板装配中,其效率不仅仅依靠怎样组织电路板本身的策略水平,而且和在该策略下的进料安排和工序安排的操作水平有关。现在需要研究的问题是怎样安排大量的电路板,以使所组成的系统整体布置安装时间达到最少。这里的问题是如何减少整体的安装时间和有效缩短

布置时间，从而和电路板整体的最优化相平衡。在这种研究中，可将布置时间和电路板的整体工作特性结合起来，使用相似的标准来加以衡量。要解决作为整体的电路板的进料分配和布置次序，一种有效的方法是基于蚁群优化的算法。评估整体装配效果和比较在不同分组算法下的结果时，仿真实验表明，这种算法对于有效缩短装配时间是非常有效的。

另外，在制造业中利用人工生命模型，能够对付现代环境中持续和不定的变化情况。人工生命不仅是用于理解自然系统的，而且适用于人造工程，如制造系统。今天，在一些生产过程优化中，已用到了原始的人工生命模型（如蚁群算法模型）。为了使该方法能够更成功地加以应用，非常有必要去研究和模拟单个智能体的适应性能力，以及群体性的智能模型。

在对白蚁筑巢行为进行研究之后，有人提出了多智能体的协作和控制策略，讨论了一种用于生产控制系统的智能方案，旨在处理各种生产变化和扰动。这种新的设计中利用了多智能体系统的概念。这个系统中的智能体使用了一种间接协调的机制，叫做Stigmergy。Stigmergy是一种传递动物间相互作用的机制，它包括间接通信，模拟的是使昆虫种群中的单个昆虫能通过局部修正引导相关环境中其他昆虫的功能。该协调机制是以蚂蚁搜索食物的技术为基础的。寻食的蚂蚁通过一种方式放出信息，并通过信息的传播使全局的信息都局部化，所以一个蚂蚁在它决策时只需通过观察它的局部环境来考虑其非局部的关系。有研究人员构造了一个原型来测试该协调技术，这个原型是基于柔性制造系统构建的。模型中包括动态顺序、或然性的处理时间和一些普通的扰动。这个原型适合于一类特殊的搜索问题，能够基于智能体目标完成一个短期的预测，并能通过简单方法去了解系统中的运行情况。另外，提出者还进一步研究了随着原型大小和复杂性的逐渐增大所带来的情况变化。

在基于蚁群算法的多目标及时生产时序问题求解中，有论文提出了一种用于生产时序问题安排的新的蚁群算法。该算法用两个目标指导模拟真实蚂蚁的人工智能体，从而得到生产时序问题的最佳解。这两个目标是生产设备的最小化和物料使用率的最优化。这类问题是具有 NP 难度的问题，因此，要得到所有解是不现实的。因此，可将具有较小运算代价的算法用于这类问题。用蚁群算法得到的结果和用其他启发式搜索方法，如模拟退火、禁忌搜索、遗传算法和人工神经网络等方法得到的结果进行比较，表明蚁群算法在性能和对 CPU 要求上相对于其他的方法更具有竞争力。

3.8.3 工程设计

有论文提出了应用蚁群系统来优化门级组合逻辑电路的设计方法。在部分电路中，通过计算蚁群系统所需要的求解距离来定义电路质量改进的度量，用最少的门级电路能实现同样电路功能的这些解被认为是最佳的。

另外，进化和自适应搜索可用于整个系统工程设计和策略优化，应用进化和自适应多级搜索策略的离散和连续变量参数蚁群算法可用于多层完整系统的初步设计。与描述工程系统主要原理的初步设计软件结合起来，这些策略能提供相关领域高水平的决策支持。其初始化工作包括具有自适应变异功能的结构化遗传算法，其中的变异功能可用来保证搜索的多样性。另外，针对结构化遗传算法的缺点，其替换策略可用包含蚁群搜索和进化搜索原理的双因子策略，这种方法已被用于离散和连续参数集的优化问题的求解。通过两个搜索因子间的合理通信，能得到比结构化遗传算法更好的搜索效率，同时这样也能简化相关的染色体描述。就复杂度而言，这种简化能使初步设计层次进一步提高复杂度，因此，这种技术对多级、离散和连续混合参数设计领域中突出的配置设计问题的研究具有重大意义。

在整个受约束系统详细设计时，进化和自适应搜索的不同策

略研究中可包括进化计算的蚁群算法模型。该模型包含相互合作的结构特点,这些结构在系统设计和有约束优化时能提供决策支持和优化能力,而系统设计的目标是为大工程系统提供最佳的初始配置。相关论文中还引入了包含细节设计的进化技术策略。这些领域的每一个方面都代表了进化/自适应搜索过程的特定问题。这里的总体目标是确定设计难点的主要区域,并提供解决综合问题的解。

基于蚁群算法,还可进行键盘排列问题的最优化设计。为了解决计算机键盘上字母的排列问题,有论文中介绍了一种新的键盘,并采用人造环境学的标准作为评估的准则。这种最优化问题被叫做键盘的排列问题。文中开发了一种蚁群优化算法,该算法相对于多种语言都能得到新的高效率的键盘排列,文中将所得结果与标准的手动键盘进行了比较研究。

3.8.4 其他工程应用领域

在露天采矿边坡临界滑动面搜索的蚁群算法研究中,可通过改进的蚁群算法结构和蚂蚁转移概率的确定方法,构造一个适应复杂边坡临界滑动面搜索功能的启发式蚁群算法。相关论文中还研究了极度干旱地区露天采矿场岩体的物理力学特性,探讨了启发式蚁群算法在采矿边坡临界滑动面搜索和稳定性分析评价中的应用。结果表明,与传统方法相比,该方法具有明显的优越性。

为了对上海市的内河航道提供科学合理的规划,解决集装箱集散问题,同时也为推广蚁群算法的应用,有论文在介绍蚁群算法解决旅行商(TSP)问题的模型之后,对蚁群算法做了相应的改进,配合随机分布技术,以上海市整个内河航道和集装箱运输为研究对象,对内河航道进行规划,得出了上海市内河集装箱集散系统合理的分配方案,并提出了为满足该合理系统所需进行的相应的河道改造策略。

在设施的布局问题求解中,也可利用此类组合启发式最优化

技术——蚁群算法。这种算法本身是受到蚂蚁集体行为启发的，因此相关论文中模拟了蚂蚁的整体组织行为，使之适合于解决该类问题。文中所给出的特定执行策略中，将禁忌搜索法作为蚁群算法的一个部件。对于一些大范围的布局问题，该方法表现出全局最优特性，能得到最好的布局方法。相关研究还表明，应用蚁群算法的布局代价可以通过局部搜索技术得到改进，改进的程度取决于所使用的技术。文中证明，蚁群算法对于解决大范围的设施布局问题是一种非常有效和可行的方法，但像其他新的适应性方法（如遗传算法）那样，它也会花费大量的计算时间。另外，蚁群算法在地下房屋建造顺序优化上也得到了运用。对于大量的地方房屋建造顺序问题求解，结果表明，这种方法是有效的。

最近 10 年，在水利分配系统的最优化设计中，进化算法（如遗传算法）得到了广泛的应用。最近，一种改进的蚁群优化算法也被用于水利分配最优化问题的设计中。这种算法被用于两个基准的水利分配系统最优化问题，所得到的结果和利用遗传算法得到的结果进行了比较。结果表明，改进的蚁群优化算法相对于遗传算法，在水利分配最优化问题求解中具有更好的应用前景。在考虑计算的效率和达到全局最优解的能力上，改进的蚁群优化算法也比遗传算法更强一些。

尽管已有许多模型被应用到了环境领域相关的研究和管理问题中，但是要得到所需的模型参数，这仍然是一个具有挑战性的工作。为了得到模型的参数，逆向建模的方法是比较诱人的。在这种方法中，可利用所得到的易测量（模型输出）去估计模型中的未知参数。这种逆向过程中，通常需要对目标函数进行最优化。在相关研究中，可以模拟蚁群的行为，用蚁群算法来求取此参数的最优解。这就是利用蚁群算法来估计不饱和土壤水利参数的总体思路。可以证明，这种通过蚁群智能来得到最优化参数的方法是很有前途的。在八种不同的应用中，蚁群算法都能够估计出正确的

相关参数。在相关的研究中,把这种方法应用于测渗计实验,用以估计水压参数。从迄今得到的结果看是非常好的,但是如果要利用蚁群优化算法来估计不稳定流量和传输参数,该方法还需要进一步改进。

在大规模的经济分配问题研究中,还有人提出了一种新的解决国家经济分配问题的协作智能体搜索方法,其主要目标之一是研究可选择因子搜索方法在系统优化中的可行性。所提出的蚁群搜索算法是基于真实蚂蚁信息素形成模式和生物学的理论结果之上的。所提出的一种新的编码技术能够克服蚁群搜索算法在连续空间搜索中所遇到的困难,并适用于经济分配问题的求解。同时,还开发了一种面向对象的改进型蚁群优化系统,并在40多个单元的实际系统中加以了应用。通过与传统遗传算法比较,结果表明,这种新出现的方法在解决一些工程问题时具有极大的吸引力。这种方法的能力来自于蚁群的大规模并行求解模式和信息的正反馈。

蚁群算法还被用于解决反应堆再生器的多目标优化问题。在具有移动减催化剂的管状反应堆再生器的多目标优化算法研究中,其任务是找到最佳的管状反应堆温度曲线图、催化剂循环率和再生能力,从而使加工过程的收益量、选择性、转化率等指标最大化。蚁群算法这种新的启发式寻优算法可以被用于搜索其最佳的组合解。

3.9 蚁群算法的应用特征

如前所述,蚁群算法这种求解模式能将问题求解的快速性、全局优化特征以及有限时间内答案的合理性结合起来,并且能够与旅行商问题的求解进行直接的对应,因此对于能够直接转化为路径优化问题的组合类寻优问题求解,应用蚁群算法是十分有效的,如交通、机器人路径寻优、电力系统路径优化配置、制造路径优化、

通信路由选择、计算机网络管理等。如果将其进行一定程度的拓展，蚁群算法也可被用于约束优化和二次配置问题的求解。另外，许多类型的流程调度问题也是典型的组合优化类寻优问题，也是适合用蚁群算法来进行求解的。本章中，作者从优化问题求解、交通、计算机、机器人、电力系统、通信、化工及其他工程应用领域出发，全面综述了蚁群算法在其中的应用成果。对于蚁群算法的优化问题求解过程，其本质在于：

（1）合理的选择机制。信息量越大的路径，被选择的概率越大。

（2）合理的更新机制。路径上面的信息量会随蚂蚁的经过而增长，同时也会随着时间的推移逐渐减少。

（3）合理的协调机制。蚂蚁之间实际上是通过信息量来进行互相通信和协同工作的，这样的自组织机制使得蚁群算法具有很强的发现较好解的能力。

但是，不可讳言，蚁群算法也有一些缺陷，例如：由于蚁群中多个个体的运动具有随机性，当群体规模较大时，要找出一条较好的路径，就需要较长的搜索时间。因此，在蚁群优化问题的求解研究中，所需要做的工作主要是合理运用其答案选择、信息更新和群体协调机制，尽量避免其算法缺陷，同时针对各种不同的优化问题求解领域，设计不同的蚁群算法来加以解决。

第4章　蚁群算法的具体描述及改进

4.1　基本蚁群算法思路

基本的蚁群算法是与旅行商问题的求解联系在一起的。我们下面要定义的系统是从实际蚁群的研究中引申出来的模型，称之为蚁群系统，相应算法称为蚁群算法。

首先，让我们再详细回顾一下第1章中所举的蚁群路径搜索实例。

我们所定义的蚁群系统与自然界中的生物蚁群系统的主要区别是：

（1）人工蚁有一定的记忆功能；

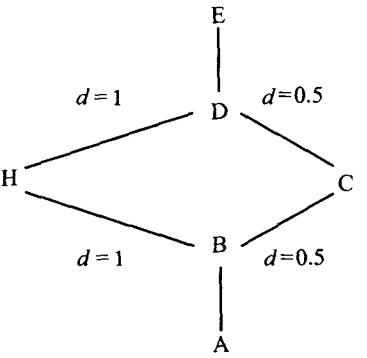
（2）人工蚁不是全盲；

（3）人工蚁处于时间离散的环境中。

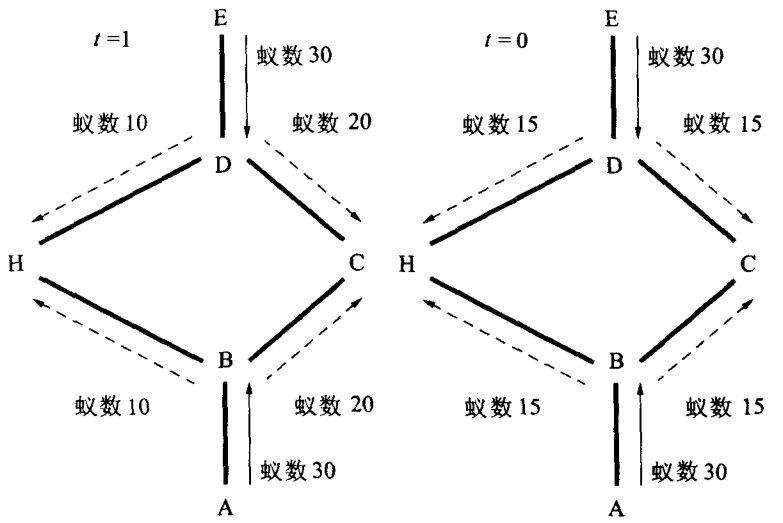
这样，所定义的蚁群算法就很适用于离散优化问题的求解。

考虑图4-1(a)，它是图1-1(b)情况的一种蚁群系统模型。假设D和H之间、B和H之间以及B和D之间(通过C)的距离为1，C位于D、B路径的中央[见图4-1(a)]。现在我们考虑在等间隔离散时间点($t=0,1,2,\cdots$)的蚁群系统情况。假设每单位时间有30只蚂蚁从A到B，另30只从E到D，其行走速度都为1(一个单位时间所走距离为1)，在行走时，一只蚂蚁可在时刻t留下浓度为1的信息素。为简单起见，设信息素在时间区间$(t+1,t+2)$的中点$(t+1.5)$时刻瞬时完全挥发。在$t=0$时刻无任何信息素，但分别有30只蚂蚁在B、30只蚂蚁在D等待出发。它们选择走哪条路径是完全随机的，因此在两个节点上蚁群可各自一分为二，走两个方向。但在$t=1$时刻，从A到B的30只蚂蚁在通向H的路

径上[图4-1(b)]发现一条浓度为15的信息素,这是由15只从B走向H的先行蚂蚁留下来的;而在通向C的路径上它们可发现一条浓度为30的信息素路径,这是由15只走BC路径的蚂蚁所留下



(a)



(b) (c)

图4-1 人工蚁群路径搜索实例

的气息与15只从D经C到达B的蚂蚁留下的气息之和[图4-1(c)]。这时,选择路径的概率就有了偏差,向C走的蚂蚁数将是向H走的蚂蚁数的2倍。对于从E到D来的蚂蚁也是如此。

这个过程一直会持续到所有的蚂蚁最终都选择了最短路径为止。

这样,我们就可以理解蚁群算法的基本思想:如果在给定点,一只蚂蚁要在不同的路径中选择,那么,那些被先行蚂蚁大量选择的路径(也就是信息素留存较浓的路径)被选中的概率也更大,较多的信息素意味着较短的路径,也就意味着较好的问题解答。

4.2 用于求解旅行商问题的蚁群算法定义

这里,我们使用通用的旅行商(TSP)问题作基准进行论述,这样可以很容易地与其他启发式寻优算法进行比较。虽然该模型定义要受所求解问题结构的影响,但从后面的论述中,我们将看出,该方法可以很容易地拓展到其他优化问题的求解中。

TSP问题属于一种典型的组合优化问题。其定义是:给定n个城市的集合,TSP问题等价于寻找一条只经过各城一次的具有最短长度的闭合路径。设d_{ij}为城i和j之间的距离;欧几里德空间中,$d_{ij}=\{(x_i-x_j)^2+(y_i-y_j)^2\}^{\frac{1}{2}}$。一个TSP问题可由图$(N,E)$给定,其中$N$是城市集合,$E$是城市之间的支路集合(欧几里德空间中TSP意义下的一个全连接图),令$b_i(t)(i=1,2,\cdots,n)$为在t时刻i城市上的蚂蚁数,则$m=\sum_{i=1}^{n}b_i(t)$为总的蚂蚁数。每只蚂蚁都可认为是具有下列特征的简单智能体:

(1) 其选择城市的概率是城市之间的距离和连接支路上所包含的当前信息素余量的函数;

(2) 为了强制蚂蚁进行合法的周游,直到一次周游完成时,才允许蚂蚁游走已访问过的城市(这可由禁忌表来加以控制);

(3) 当完成一次周游,它在每条访问过的支路(i,j)上留下一种叫信息素的物质;

令$\tau_{ij}(t)$为支路(i,j)在时刻t上的信息素强度。每只蚂蚁在t时刻选择下一个城市,并在$t+1$时刻到达那里。因此,若我们称由m只蚂蚁在区间$(t,t+1)$内做的m次移动为AS算法的一次迭代,则算法每迭代n次(我们称为一个周期),每只蚂蚁就完成了一次周游。在这个时间点,信息素强度可根据下面公式进行更新:

$$\tau_{ij}(t+n) = \rho\tau_{ij}(t) + \Delta\tau_{ij}$$

$1-\rho$代表了t时刻和$t+n$时刻之间信息素的挥发程度,

$$\Delta\tau_{ij} = \sum_{k=1}^{m}\Delta\tau_{ij}^{k}$$

$\Delta\tau_{ij}^{k}$为第k只蚂蚁在t时刻与$t+n$时刻之间留在支路(i,j)上的单位长度的信息素量(对真实蚂蚁而言为气息)。可按下式进行计算:

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{L_k}, & \text{若第 } k \text{ 只蚂蚁在其周游过程的 } t \text{ 和 } t+n \\ & \text{时刻之间经过支路} \\ 0, & \text{其他} \end{cases}$$

式中,Q为常数,L_k是第k只蚂蚁的周游长度。系数ρ必须设为一个小于1的数,以避免信息素无限制地积累。在具体仿真实验中,为避免计算机出错,我们把0时刻的信息素强度$\tau_{ij}(0)$设为一个小的正常数c。

为满足一只蚂蚁访问所有n个城市的约束,我们将每只蚂蚁与一个称为禁忌表的数据结构联系起来,该表存储了直至时刻t的所有已访问城市,并禁忌蚂蚁在n次迭代(一个周游)结束之前再次访问它们。一次周游结束时,该禁忌表可被用来计算蚂蚁的当前解(也即蚂蚁所走路线的长度)。计算之后可清空禁忌表,这时蚂蚁又可以自由选择路径了。我们定义Tabu_k为包含第k只蚂

蚁禁忌表的动态增长矢量，$tabu_k$ 为从 $Tabu_k$ 的元素得到的集合，$tabu_k(s)$ 为表中的第 s 个元素（也即第 k 只蚂蚁在当前游走中访问的第 s 个城市）。

我们称 $\frac{1}{d_{ij}}$ 为能见度 η_{ij}。这个量在蚁群算法运行过程中不作修改。这与刚才按式(1)计算的信息素不同。我们定义蚂蚁 k 从 i 城到 j 的转移概率为：

$$P_{ij}^{k}(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}\cdot[\eta_{ij}]^{\beta}}{\sum_{k\in\{\text{允许}k\}}[\tau_{ik}(t)]^{\alpha}\cdot[\eta_{ik}]^{\beta}}, & \text{若 } j\in\{\text{允许 } k\}\\ 0, & \text{其他}\end{cases}$$

其中允许 $k=\{n-tabu_k\}$，α 和 β 为控制信息素和能见度之间相对重要性的参数。因此，转移概率是能见度（意味着近的城市被选中的概率大，这样就执行了贪婪性试探法）和 t 时刻信息素浓度之间的折衷方案[意味着若在此支路(i,j)上有很大交通量时，那么这条支路是很有吸引力的，这样就执行了自催化过程]。

根据前面的定义，蚁周算法可以表述如下：

在零时刻，首先进行的是初始化步骤，将蚂蚁安置在不同的城市，同时设置支路上信息素的初始值 $\tau_{ij}(0)$，所有蚂蚁禁忌表内第一个元素被设成它的出发点。此后，每只蚂蚁从城 i 向城 j 游走，它选择城市的概率是两个满意度指标的函数。其中，$\tau_{ij}(t)$ 表明过去有多少蚂蚁选择了相同的支路(i,j)，而能见度 η_{ij} 意味着城市越近就越可取。若设 $\alpha=0$，就意味着不再考虑信息素水平，这样就得到了有多重起点的随机贪婪算法。

在经过 N 次迭代后，所有的蚂蚁都完成了一次周游，它们的禁忌表将全满。此时已算得每只蚂蚁 k 的 L_k 值，这样，$\Delta\tau_{ij}^{k}$ 的值也得到了更新。同时，可以存储蚁群所找到的最短路径（即 $\min L_k$，$k=1,2,\cdots,m$），清空所有的禁忌表。这个过程可一直迭代到周游计数器达到最大值（由用户定义）NC_{max}，或所有蚂蚁都在作相同

的周游。

我们将后一种情况称为停滞行为,因为它表示算法停止了可选解的搜索。

蚁周算法的规范表示如下:

(1) 初始化:$t:=0$ {t 为计数器}

$NC:=0$ {NC 为周期计数器}

给每条支路(i,j)设置信息素强度初值 $\tau_{ij}=c,\Delta\tau_{ij}=0$

把 m 个蚂蚁放在 n 个节点上

(2) $s:=1$ {s 是禁忌表下标}

for $k:=1$ to m do

把第 k 只蚂蚁的起始城市位置存入 $\text{tabu}_k(s)$

(3) 循环执行以下步骤,直至禁忌表全满{该步要循环$(n-1)$次}

$s:=s+1$

for $k:=1$ to m do

以概率 $P_{ij}^k(t)$ 选择下一个城市 j,其概率具体由式(4)给定{在 t 时刻,蚂蚁 k 在 $i=\text{tabu}_k(s-1)$ 城}

把蚂蚁 k 移到城市 j

在 $\text{tabu}_k(s)$ 中插入城市 j

(4) for $k:=1$ to m do

把蚂蚁 k 从 $\text{tabu}_k(n)$ 移回到 $\text{tabu}_k(1)$

计算蚂蚁 k 所走周游的长度

更新所找到的最短周游路线

for 每条支路(i,j)

for $k:=1$ to m do

$$\Delta\tau_{ij}^k=\begin{cases}\dfrac{Q}{L_k}, & \text{若}(i,j)\in \text{tabu}_k \text{ 所描述的周游}\\ 0, & \text{其他}\end{cases}$$

(5) for 每条支路(i,j)计算 $\tau_{ij}(t+n)$

根据方程 $\tau_{ij}(t+n)=\rho\tau_{ij}(t)+\Delta\tau_{ij}$

设 $t:=t+n$

设 $NC:=NC+1$

for 每条支路(i,j)设 $\tau_{ij}=0$

(6) 若$(NC<NC_{max})$且(无停滞行为)

则清空所有禁忌表

返回(2)

否则 打印最短路径

停止

如果我们在 NC 次后终止该算法,则蚁周算法的复杂度为 $O(NC\cdot n^2\cdot m)$。事实上,步骤(1)的计算复杂度为 $O(n^2+m)$,步骤(2)的计算复杂度为 $O(m)$,步骤(3)的计算复杂度为 $O(n^2\cdot m)$,步骤(4)的计算复杂度为$O(n^2\cdot m)$,步骤(5)的计算复杂度为$O(n^2)$,步骤(6)的计算复杂度为$O(n^2\cdot m)$。总算法的计算复杂度为 $O(NC\cdot n^3)$。

在蚁群算法的另两个版本:蚁密和蚁量算法模式下,信息素更新的方式上是有差异的。在这两种模型中,每只蚂蚁在每一步后都留下了它的信息素,而不必等到周游结束。在蚁密算法中,蚂蚁每次从 i 到 j 都会在支路(i,j)上留下数量为 Q 的痕迹;在蚁量算法中,一只从 i 到 j 的蚂蚁在支路(i,j)上留下数量为 Q/d_{ij} 的痕迹。

故而在蚁密算法中,我们有:

$$\Delta\tau_{ij}^{k}=\begin{cases}Q,\text{若蚂蚁 }k\text{ 在时刻 }t\text{ 到 }t+1\text{ 之间从 }i\text{ 到 }j\\0,\text{其他}\end{cases}$$

在蚁量算法中,我们有:

$$\Delta\tau_{ij}^{k}=\begin{cases}Q/d_{ij},\text{若蚂蚁 }k\text{ 在时刻 }t\text{ 到 }t+1\text{ 之间从 }i\text{ 到 }j\\0,\text{其他}\end{cases}$$

从这些定义可以很清楚地看到,在蚁密算法中,当一只蚂蚁从 i 到 j 时,支路 (i,j) 上的信息素的增强与 d_{ij} 无关;而在蚁量算法中,信息素的增强与 d_{ij} 成反比,即短路径对蚁群而言更有吸引力。

在以旅行商问题为代表的组合优化类问题求解中,蚁群算法体现了其非常优越的求解特征:

(1) 通用性。它可用于解同一类型的优化问题,从 TSP 问题到 ATSP 只需作直接扩展即可。

(2) 鲁棒性。只要作很小的改动,就可将蚁群算法用于解其他组合优化问题,如二次分配问题和作业安排调度问题。

(3) 群体性。这是一种基于群体的方法很有趣,因为它允许采用正反馈作为搜索机制。

(4) 并行性。该算法适用于并行操作,在求解大规模的优化问题时,不仅可以从算法本身的优化出发来提高求解效率,也可以从算法的执行模式出发来进行研究。

4.3 蚁群算法的改进思路

从一系列实验结果表明,蚁群算法具有很强的发现较好解的能力,并且不容易陷入局部最优。这是因为,该算法不仅利用了正反馈原理,在一定程度上加快了进化过程;而且该算法是一种本质并行的算法,个体之间不断地进行着信息交流和传递,这就有利于较好解的发现。虽然其单个个体容易收敛于局部最优,但多个个体通过合作,会很快收敛于解空间的某一子集。但是,这种算法也存在着一些缺陷,如:需要较长的搜索时间,当问题规模较大时也易陷入局部最优解,即产生过早收敛等问题。蚁群中智能个体的运动是随机的,在算法进化的初级阶段,各个路径上的信息量相差不明显,通过信息正反馈,使得较好路径上的信息量能够逐渐增大;但当群体规模较大时,就很难在较短的时间内从大量杂乱无章的路径中找出一条较好的路径。

作为一种体现群体协作寻优特征的启发式优化方法，蚁群算法的改进模式无非可以从以下两方面进行考虑：

（1）算法的总体结构：在这里，需要考虑的是蚁群的总体结构和组织模式。当然，各子系统间的信息联系模式也需要考虑。如多群体蚁群算法和混合型蚁群算法等，就是由几个蚁群来协同解决 TSP 问题的算法，而传统的蚁群算法中只有一个蚁群。而且，在这些算法的信息联系模式中，蚁群群体层的交互还使用了正负两种信息素效应。由于引入了群体层的交互作用，蚁群能更好地交换问题解决过程中的规划信息，并保持它们在搜索过程中的多样性。

另外，还可考虑不同信息交换方式的多重蚁群算法和具有分工特征的蚁群算法等改进方向。

（2）算法的具体参数设定和调整策略：在这里，可以对参数的选择和变化模式进行设定，如最大最小蚁群算法的使用就形成了比传统蚁群算法更贪婪的搜索模式。另外，带变异特征的蚁群算法、Ant-Q 算法、带禁忌搜索策略的蚁群算法等，也在具体的算法参数优化方面取得了一定的效果。如果在基本蚁群算法中引入变异机制，对参数进行变异式改进，则可使算法既有较快的求解速度又有较高的求解精度。而自适应蚁群算法、动态蚁群算法则是一种分布平衡的蚁群算法，可以在加速收敛和防止早熟、停滞现象之间取得很好的平衡。如果将随机特征引入算法参数的修正模式，则所形成的随机扰动蚁群算法必然具有很强的全局搜索能力。如果对算法关键参数进行分配策略和修正策略研究，采用基于目标函数值的信息素分配策略等，则可根据目标函数值来自适应调整蚁群的搜索行为，从而保证算法快速找到全局最优解。当然，也可将其他的启发式算法用于信息素分配，以及搜索过程中最优解的筛选等过程，并且与其他智能算法相结合，形成一些效果较好的改进型蚁群算法方法。在此领域的研究工作是很有前途的，同时也

是很不完善的。

4.4 蚁群算法改进实例

4.4.1 最大最小蚁群算法

最大最小蚁群算法(MMAS)是由德国学者 T. Stuetzle 和 H. Hoos 提出的,这种改进型蚁群算法的性能与传统蚁群算法的共同点是:在算法的每次迭代中只允许表现最好的蚂蚁更新路径上的信息素,该算法目的主要在于防止过早的算法停滞现象。他们的基本做法是:限定信息素浓度允许值的上下限,并且采用了平滑机制。

算法过早的停滞(早熟)现象出现是因为信息素过分集中到几条较好的路径上,而其他路径则由于长时间没有蚂蚁经过,信息素逐渐挥发,致使路径上的信息素浓度趋于0。这样,蚂蚁就几乎不会再选择这些信息素浓度极低的路径,从而丧失了探索新路径的可能,算法就表现出停滞的特征。为了减少在搜索早期算法出现停滞的可能,可以对算法中信息素浓度引入最大值和最小值限制,将每条路径上的信息素 $\tau_{ij}(t)$ 限定在 $[\tau_{\min},\tau_{\max}]$ 之间,即 $\tau_{\min} \leqslant \tau_{ij}(t) \leqslant \tau_{\max}$。在每次迭代之后,都保证信息素浓度满足该式。如果 $\tau_{ij} > \tau_{\max}$,则 $\tau_{ij} = \tau_{\max}$;同样,如果 $\tau_{ij} < \tau_{\min}$,则 $\tau_{ij} = \tau_{\min}$。当然,同时还应该保证 $\tau_{\min} > 0$,并且,如果对于所有的解元素有 $\eta_{ij} < \infty$,那么选择一特定解元素的概率不为0。

在搜索过程中,可以由下式设置信息素的最大值 $\tau_{\max}$ 为一个最大极限值的估计,其中 s^{gb} 为全局最优解:

$$\lim_{t\to\infty}\tau_{ij}(t) = \tau_{ij} \leqslant \frac{1}{1-\rho}\cdot\frac{1}{f(s^{gb})}$$

当得到最优解时,按照该公式更新 $\tau_{\max}$,这样便可得到一个动态的变化值 $\tau_{\max}(t)$。

为了决定 $\tau_{\min}$,可以使用下面的假设:

(1) 最优解是在搜索停滞前得到的。在这种情况下,算法的一次迭代中找到全局最优解的概率就远远大于0,因为较好的解可能就处于接近最优解的地方。

(2) 影响解的一个主要原因取决于信息素最大和最小值限制的差异,而不是启发式信息。

根据上面的假设,可以选择一个好的 $\tau_{\min}$ 值来提高算法的收敛性。这里先给定一个 P_{best} ($P_{\text{dec}} = \sqrt[n]{P_{\text{best}}}$),然后根据下式设置 $\tau_{\min}$:

$$P_{\text{dec}} = \frac{\tau_{\max}}{\tau_{\max} + (\bar{\tau} - 1)\tau_{\min}}$$

式中,$\tau_{\min} = \dfrac{\tau_{\max}(1 - P_{\text{dec}})}{(\bar{\tau} - 1)P_{\text{dec}}} = \dfrac{\tau_{\max}(1 - \sqrt[n]{P_{\text{best}}})}{(\bar{\tau} - 1)\sqrt[n]{P_{\text{best}}}}$

如果 $P_{\text{best}} = 1$,那么 $\tau_{\min} = 0$。如果 P_{best} 太小,通过上式计算得到的值可能会出现 $\tau_{\min} > \tau_{\max}$,这时可设置 $\tau_{\min} = \tau_{\max}$。

信息素浓度最大值和最小值的选择取决于平均路径的长度。通过设定信息素的上下限,不会使某条路径的信息素浓度过高,从而导致过早停滞,也不会因为某些路径的信息素浓度过低而导致算法发现新路径可能性的降低。这样,通过对信息素浓度的限制,就降低了算法搜索中的早期收敛(停滞)问题。特别是,对于要求迭代次数较多的问题求解过程,这样的寻优模式将具有更好的效果。

在最大最小蚁群算法中,只允许其中的一个路径更新信息素。该路径通常是最优路径,它可以是在所有周游过程中已经找到的最优路径,也可以是在当前周游中找到的最优路径。信息素可按照下面的公式进行更新:

$$\tau_{ij}(t+1) = \rho\tau_{ij}(t) + \Delta\tau_{ij}^{\text{best}}$$

式中,$\Delta\tau_{ij}^{\text{best}} = 1/f(s^{\text{best}})$,并且 $f(s^{\text{best}})$ 表示该次迭代中的最优

路径(s^{ib}),或者是全局最优路径(s^{gb})的代价函数。仅使用一只蚂蚁来更新所找到路径的方法在传统蚁群算法中也可以应用,不过在传统算法中,使用的仅是(s^{gb})[尽管很有限的试验使用了(s^{ib})]。当仅使用(s^{gb})对信息素进行更新时,搜索可能会过快地收敛,对可能的较好解的范围进行开发的过程也将受到限制,这将会导致算法的局部收敛。如果只使用(s^{ib}),则可避免上述情况。因为,该次迭代的最优路径(解)会随着迭代次数的不同而显著变化,并且会有较大数量的路径(解)得到增强。当然,这里还可以使用混合策略,比如使用(s^{ib})来作为信息素更新的默认值,而仅在固定的迭代次数间隔时使用(s^{gb})。

事实上,当我们使用最大最小蚁群算法,并结合局部搜索算法来解决大规模的 TSP 问题或者二次配置经典实例时,最好的策略应该是使用一种动态的混合策略,即:在搜索的过程中,增加使用(s^{gb})的频率来对信息素进行更新。这样,在初始化时,每条边上的信息素 $\tau_{ij}(0)=\tau_{\max}$;在每次迭代后,信息素浓度以 ρ 挥发,并且只有找到最优路径的蚂蚁才能增加它的信息素浓度,使其保持较高的水平。因此,较差路径上的信息素浓度就会增加较慢,而只有最优路径上的信息素浓度才会维持一个较高的水平,并且被更多的蚂蚁所选择。

这样,最大最小蚁群算法相对于基本的蚁群算法,其性能必然有相当大的提高。但是这里需要指出的是,尽管在算法中对信息素浓度进行了最大和最小值的限制,但最大最小蚁群算法仍然可能出现停滞现象。也就是说,只采用最大最小痕迹浓度的限制还不足以在较长的运行时间里持久消除停滞现象。因此,这里可采用信息素的平滑机制:按照比例更新的方法来调整信息素的浓度,使信息素浓度的增加值正比于 $\tau_{\max}$ 与边 (i,j) 上的信息素浓度 $\tau_{ij}(t)$ 的差值,即:

$$increase \propto \tau_{\max} - \tau_{ij}(t)$$

在最大最小蚁群算法中，还可结合局部搜索算法。局部搜索算法是从一个初始解 $i \in S$ 开始的，然后运用一个产生器，持续在解 i（当前解）的邻域 S_i 中搜索比 i 更优的解。若找到比 i 更优的解，就用这个解取代 i，成为当前解，再继续计算；否则算法终止，当前解就是算法的最终解。在该算法中可使用局部搜索算法的原因有两点：一方面，通过结合局部搜索模式，可以提高算法的性能，从而可以在早期找到较好的解，并且能更直接地引导学习机制；另一方面，最大最小蚁群算法还可以为随后的局部搜索阶段构造一个好的初始路径，这样就可以逐步找到一个接近最优的路径。蚁群算法与局部搜索方法相结合，就提高了局部搜索的效率。实验也表明最大最小蚁群算法在寻找解的有效性方面和防止算法的过早停滞方面有了较大改进。

该算法的一个不足之处在于它运行时间较长。当算法在每次迭代中构造 m 条路径，并且每条路径的计算复杂度为 $O(n^2)$ 时，最大最小蚁群算法对于每一次迭代，其总的计算复杂度就为 $O(m \cdot n^2)$。通常我们可以选择 m 等于节点的数量（在 TSP 问题中就是城市的数量），那么算法的总体复杂度就为 $O(n^3)$。因此，运行时间就会随着问题规模的增加而大大增加。为了降低运行时间，可以考虑增加候选解集合的选择概率。这样，除了可以降低算法的运行时间之外，还可以增加该算法的性能。这些改进模式可以通过对一些较大规模的寻优问题求解加以证明。

4.4.2 具有变异和分工特征的蚁群算法

4.4.2.1 对选择策略的改进

在蚁群算法中，在蚂蚁搜索过程的起始阶段，有的路径上有蚂蚁走过，有的路径还未来得及被走过。而蚂蚁选路的策略是，一旦有路径上的信息素（即信息量）多于其他路径，它就以较大的概率选择该路径。这样就使得蚂蚁从搜索的一开始就以较大的概率集中在几条当前局部长度较短的路径上。为了避免蚁群一开始就失

去解的多样性，在路径上信息量未达到一定阈值时，让蚂蚁忽视较优解的存在。只有当信息量的刺激趋于所设阈值 ρ_0 时，才让蚂蚁在信息量的刺激下趋于信息量累计较多的路径。这样，各蚂蚁智能体就可在寻优的初始阶段选择较多的路径，以保证解的多样性。

这里，可以让第 k 只蚂蚁按以下概率从状态 i 转换到状态 j：

$$j = \begin{cases} \max\{\tau_{is}^{\alpha}, \eta_{is}^{\beta}\}, & s \in allowed_k, \text{ 若 } r \leqslant \rho_0 \\ \text{依概率 } P_{ij}^{k} \text{ 选择 } j, & \text{其他} \end{cases}$$

式中，$\rho_0 \in (0,1)$，r 是(0,1)中均匀分布的随机数，由此便增加了所得解的多样性，并在一定程度上削弱了蚁群陷入局部最优的趋势。

4.4.2.2 蚁群信息量的全局修正

信息量的全局修正规则如下：

$$\tau_{ij}(t+n) = \rho\tau_{ij}(t) + (1-\rho)\Delta\tau_{ij}$$

式中，$\Delta\tau_{ij} = \Delta\tau_{ij}^{k}$，其中的第 k 只蚂蚁是发现本次循环中最短路径的蚂蚁，$\Delta\tau_{ij}^{k}$ 按经典算法中的公式求得。

在全局修正规则中，只让实现最好周游的蚂蚁释放信息素。它和改进的状态转移规则相结合的搜索模式，保证了蚂蚁在优秀父辈完成的周游领域内进行更多搜索，这就使得相应的求解速度大大提高。

4.4.2.3 引入变异

为了克服蚁群算法计算时间较长的缺陷，这里可以引入遗传算法中变异算子，经过局部优化后，整个群体的性能就会有明显改善，使算法保持更好的多样性特征。蚁群算法的变异方法主要有两类：

(1) 逆转变异：在所得的某一解中随机选两点，再把这两点内的路径按反序插入原位置中。例如，路径 $A = \{012\dot{3}45\dot{6}7890\}$ 的逆转点选择3、6，经逆转后，变为 $A^* = \{01265437890\}$。这种变

异操作对于 TSP 问题,就调整前后的 TSP 圈的长度变化而言属于非常细微的调整,因而可使得局部优化的精度提高,但所需的计算稍微复杂一些。在实现过程中,逆转变异法可在每次循环安排完 m 只蚂蚁的路径之后、信息素调整之前进行,且应考虑到时间问题。在具体的应用中,有的方案只对每次循环的最优路径进行变异,如果变异后路径长度小于变异前,就保留这个最优解,否则就维持路径原值。

(2) 插入变异:变异方法是从所得路径中随机选择一个码,并将此点插入随机选择的插入点中。比如,对上面的 A 路径,如果选择插入码为 5,选取插入点为 2 ~ 3 之间,则经插入变异后,$A^* = \{01253567890\}$。在具体的应用中,可在蚁群算法循环全部完成后对所求得的全局最优解进行变异。且为了达到最优效果,对最优路径可能进行的每一种插入都应进行尝试,如果所得解有进展就加以保留,否则就维持原值。

4.4.2.4 蚁群分工

受蚁群筑巢模式的启发,蚁群中的蚂蚁从事不同的工作,即蚂蚁之间是互相分工的。在蚁群算法的改进中对各蚂蚁智能体进行分工,我们称之为具有分工的蚁群算法。

如在 TSP 问题求解中,蚂蚁从不同的顶点出发,就相当于这些蚂蚁根据出发的顶点不同而进行分工,并且从自己出发的顶点找出回到自己顶点的最短路径。如果不加分工,就很难得到最短路径。这里,如果在每个顶点放上 15 只蚂蚁,就相当于分工 144 种蚂蚁,每种 15 只。另外,在函数优化中,通常其数学描述为:

$$\min f(x_i); x_{i\min} \leqslant x_i \leqslant x_{i\max}, x \in R^n, i = \{1,2,3,\cdots\}$$

如果对每个 x_i 的区间派一种蚂蚁去搜索,或者让每一种蚂蚁搜索一个变量取值范围,就可使得每种蚂蚁的搜索空间大大减少。这样,不但降低了求解问题的难度,还增强了蚁群算法的搜索能力。

4.4.3 随机扰动蚁群算法

4.4.3.1 基本原理

蚁群算法的主要依据是信息正反馈原理和某种启发式算法的有机结合，其转移概率公式就揭示了这一原理。它表明，如果放到某条路径上的信息素越多且路径越短，那么该路径被蚂蚁选中的概率就越大，类似于遗传算法中的“轮盘赌法”。鉴于这样一层物理含义，可以设计如下更为简洁的转移策略：

$$C_{ij(k)} = \begin{cases} (\tau_{ij(k)}\eta_{ij(k)})^{\gamma}, j \notin \text{tabu}_{(k)} \\ 0, \text{否则} \end{cases}$$

s 为 $\max(C_{ij(k)})$ 所对应的城市

式中，$\gamma > 0$ 为扰动因子。需要指出的是，公式中的 $C_{ij(k)}$ 不再是“转移概率”，而是路径 ij 的“转移系数”。由于蚂蚁总是选择转移系数最大的路径，这个值就具有一定的确定性。经分析可以发现，当 $C_{ij(k)}$ 取固定值时，算法与基本蚁群算法相同，仍不可避免出现停滞现象，因此有方案提出采用可变的扰动因子，考虑以下两点：

（1）蚂蚁个体的运动总是沿着转移系数最大的路径移动，当群体规模较大时，很难在短时间内从大量杂乱无章的路径中找出一条较好的封闭路径。因此在最初的几次迭代中，为加速算法的收敛，该因子应取较大的值，才能使得较好路径上的信息量明显高于其他路径上的信息量。

（2）若该因子一直不变，则必将导致某一路径上的信息量远远高于其他路径上的信息量。而此路径并不一定是最优的，这就会导致随后的搜索出现停滞现象。由此，在随后的搜索过程中应适当减小此数值。这一方面可以提高路径选择的多样性（即起到一定的扰动作用），另一方面又可以使收敛过程趋于平缓。

这里，可以设计倒指数关系曲线来描述扰动因子 γ：

$$\gamma = ae^{b/k}, k = 1,2,\cdots,M; a > 0; b > 0$$

式中，M 表示最大的迭代次数；a，b 表示扰动尺度因子。特别地，为不失随机性，令 $a = a'X$，其中 $a' > 0$，X 是(0,1)中均匀分布的随机数。上式可知，随着迭代次数的增大，γ 的值最终趋近于系数 a，而系数 b 的大小决定了曲线趋近于系数 a 的快慢程度。

考虑到传统蚁群算法中易出现的停滞现象，还可以采用随机选择策略，并在进化过程中动态调整随机选择的概率。同时，为了防止最优路径的漏选，对信息量最大的路径可以单独计算其概率。这样，就可设计如下的具有随机扰动特性的转移系数：

$$C_{ij(k)} = \begin{cases} (\tau_{ij(k)}\eta_{ij(k)})^{\gamma}, \tau_{ij(k)} = \max(\tau_{is(k)}), s \notin \text{tabu}_{(k)} \\ (\tau_{ij(k)})^{\alpha} \cdot \eta_{ij(k)}, \tau_{ij(k)} = \tau_{is(k)} - \max(\tau_{is(k)}), \\ \qquad\qquad \text{且 } U \leqslant P_m, s \notin \text{tabu}_{(k)} \\ (\tau_{ij(k)}\eta_{ij(k)})^{\gamma}, \tau_{ij(k)} = \tau_{is(k)} - \max(\tau_{is(k)}), \\ \qquad\qquad \text{且 } U > P_m, s \notin \text{tabu}_{(k)} \\ 0, \text{否则} \end{cases}$$

式中，γ 为具有倒指数的扰动因子；$P_m \in (0,1)$ 称为随机变异率；U 是(0,1)中均匀分布的随机数。

该公式表明，某次迭代过程中某只蚂蚁有若干条路径可选，对于信息素密度最大的那一条路径，可应用转移系数公式；而对于其他的可选路径，则采用随机选择方式。该公式是确定性选择与随机选择相结合的产物。确定性选择导致蚂蚁总是选择转移系数最大的路径，而随机选择导致计算转移系数时具有较强的随机性。正是两者的共同作用才使算法具有更强的全局搜索能力。

4.4.3.2 参数选取

在以上参数中，α，ρ，Q，γ，a，b 都是非常重要的，其选取对于计算的结果影响较大。参数 α 表示某一路径的信息量对蚂蚁选择路径的影响程度，参数 Q 的大小决定了路径上信息量的更新程度，对于不同的网络，它们的取值差别较大，难以确定其最佳的取

值范围。变量$\rho \cdot \tau$的物理含义为残留的信息素密度,即需要忘记一部分过去积累的信息,以便更好地利用最新的信息。所以,若ρ取值过小,则不能很好地利用过去积累的信息;若ρ很大,则不能达到信息素密度有效更新的目的。对于上述参数,目前的解析法还难以确定其最佳组合。这里,只能通过大量的数字仿真来进行参数的优化设置。

一般地,蚁群算法的参数可通过反复试凑得到,显然这将对算法的计算效率和收敛性产生不利影响。为此,可事先通过大量的数字仿真总结出一种较有效的参数选取方法,以此指导后续的参数优化工作。具体步骤如下:

(1) 经大量仿真实验发现,参数α和Q值对算法的计算进程影响较大,取值范围也较大;而参数ρ和P_m对算法的计算进程的影响相对较小,其取值也相对固定(在0~1之间)。因此,我们首先随机设定一组ρ和P_m的值(在0~1之间),来调整α和Q得到较理想的解,称之为"粗调"。

(2) 在基本确定α和Q两个值之后,反过来调整ρ和P_m,寻找更优的解,称之为"细调"。此后,在"细调"得到ρ和P_m的基础上,再进行"粗调"得到更好的α和Q值。如此反复,最终可得出一组较为理想的参数组合。

这种方法是在对不同的TSP问题进行大量数值仿真实验的基础上总结出来的,具有一定的普遍意义。

4.4.4 自适应蚁群算法

针对蚁群算法加速收敛和早熟停滞现象的矛盾,这里介绍一种基于分布均匀度的自适应蚁群算法。该算法可以根据优化过程中解的分布均匀度,动态地调整信息量更新策略和选择路径概率。这样,可以在加速收敛和防止早熟停滞现象之间取得很好的平衡。

4.4.4.1 聚度和信息权重

在蚁群算法中,加速收敛和防止早熟停滞现象是一对矛盾。

为了加速收敛，Ant-Q 算法让信息量最大的路径对每次路径的选择和信息量的更新起主要的作用，但由于强化了最优信息的反馈，就可能导致早熟停滞现象。而最大最小蚁群算法将各个路径上的信息量的更新限定在固定的范围内，这虽然在一定程度上避免了早熟停滞现象，但在解分布较分散时会导致收敛速度变慢。以上方法的共同缺点在于，它们都按一种固定不变的模式去更新信息量和确定每次路径的选择概率。在一种基于分布均匀度的自适应蚁群算法中，根据蚂蚁路径选择的分布均匀情况，动态地调整信息量更新策略和确定每次选择路径的概率。这样，可以在加速收敛和防止早熟停滞现象之间取得很好的平衡。为此，我们基于常规数学模型引入“聚度”的概念来衡量解的均匀程度，从而决定每次选择路径的概率以及信息量更新的策略。

当所有路径上蚂蚁的分布相对比较分散时，其聚度就较小，此时难以强化最优信息，并可能导致搜索速度较慢。因此，这时要强化正反馈信息，应该让较少的几个较优的路径有较大的概率被选取。在信息量更新时，应该仅让较少的几个较优的路径上的信息量得到较大程度的增强；反之，在所有路径上蚂蚁的分布相对比较集中时，聚度就较大，此时易引起早熟停滞现象。因此，这时应使解趋于多样化，使较多的路径有可能被选取。在信息量更新时，应该让较多路径上的信息量得以增强。通过这样动态的自适应调节，可以在有效改善蚂蚁搜索速度的同时避免局部优化。为此，在基于分布均匀度的自适应蚁群算法的迭代过程中，应根据各条可能的路径上的聚度来确定下一次迭代中可供蚂蚁选择的路径的信息权重，并以此来确定它们被选中的概率。此外，算法中也根据该分布范围内各条路径所构成解的优劣及信息权重，对其信息量进行有差别的动态更新，从而实现信息量分布的自适应调节。这里，仍然以求解 TSP 问题为例来说明此项方法。

定义 1 设从城市 i 共有 r 条路径到达另外 r 个城市 $i_1, i_2,$

$\cdots, i_r$，另设上一次迭代中，经过这 r 条路径上的蚂蚁数分别为 a_1，$a_2, \cdots, a_r$，令：

$$sta(i) = \sqrt{\sum_{l=1}^{r}\left(\frac{m}{r} - a_l\right)^2} \text{ 为城市 } i \text{ 的聚度}$$

若在上一次迭代中，m 只蚂蚁遍历时经过以城市 i 为起点的 r 条路径中的 s 条，设经过它们的蚂蚁个数分别为 $a_1, a_2, \cdots, a_s$，则这些值均不为0，而其余路径上的蚂蚁个数均为0。城市 i 的聚度随 s 的减小而增大，在极端情况下：

（1）当 m 只蚂蚁在以城市 i 为起始点的 r 条路径上均匀分布时，每条路上有 m/r 只蚂蚁，城市 i 的聚度为：

$$sta(i) = \sqrt{\sum_{l=1}^{r}\left(\frac{m}{r} - \frac{m}{r}\right)^2} = 0$$

（2）当 m 只蚂蚁集中在以城市 i 为起始点的 r 条路径中的某一条上时，城市 i 的聚度为：

$$sta(i) = \sqrt{\sum_{l=1}^{r-1}\left(\frac{m}{r}\right)^2 + \left(m - \frac{m}{r}\right)^2} = m\sqrt{1 - \frac{1}{r}}$$

记该值为 $\max sta(i)$，即最大的可能聚度值。

若城市的聚度较大，说明蚂蚁上一次从这个城市到达另外城市的路径集中于少数几条上，也说明信息量集中于这少数几条路径。这样，在以后的最优解搜索过程中，蚁群在该城市选择下一城市的可选路径就会相对集中，因过度强化正反馈信息引起早熟停滞现象的可能性就越大。相反，当城市的聚度越小时，这个城市到达另外城市的信息量分布就会相对比较分散，这不利于强化最优信息，会导致收敛速度变慢。为了在这两方面达到有效的平衡，以在改善蚂蚁搜索速度的同时避免局部优化，可考虑根据城市 i 的聚度 $sta(i)$ 来确定蚂蚁在该城市时下一步可供选择的路径的条数 w。这里取：

$$w = \left\lfloor \frac{sta(i)}{\max sta}(r-1) + 0.5 \right\rfloor + 1$$

在路径选择过程中,蚂蚁仅考虑信息量最高的 w 条路径。显然,当城市的聚度越大时,w 越大,蚂蚁下一步的分布范围越来越广;反之,聚度越小,蚂蚁搜索时分布范围就越小。在极端情况下,当聚度最大时,即 $sta(i) = \max sta$ 时,$w = r$,m 只蚂蚁可选择所有可选路径;当聚度最小时,即 $sta(i) = 0$ 时,$w = 1$,蚂蚁只能选择一条最优路径。

虽然这里约定了蚂蚁下一次搜索的分布范围,但是在一般的蚁群算法中,蚂蚁的选择策略主要依赖于根据其所在的当前城市 i 来选择下一城市的期望程度 η_{ij}(即可访问度,这里取一般值 $1/d_{ij}$)和以 i 为起点的各条路径上的信息量强度 $\tau_{ij}(t)$,这会在一定程度上误导大量的蚂蚁聚集于当前信息量较大的几条局部距离较短的路径上。因此,这里可引入"信息权重"这个量来限制信息量和期望程度对蚂蚁选择概率的影响程度,从而调整各个路径被选中的概率。其总体原则是要使信息量大的和当前局部距离小的路径被选择的概率大些。若信息量分布比较集中,则应当使各条路径上的信息量及期望程度对选择概率的影响的差别小些,各条路径被选中的概率相对均匀些;反之,应当使信息量和期望程度对选择概率的影响的差别大些,这样蚂蚁选择的路径会相对更集中一些。

定义 2 以城市 i 为起点的 r 条路径按其信息量由高到低排序,并将序号依次存于数组 $rank$ 中,即数组元素 $rank[j]$ 的值为路径 (i,j) 的序号。下一步可供选择的路径条数为 w,这里,可取 $q = w/r$,记:

$$\xi_{ij} = \begin{cases} q^{rank[j]-1}, \text{如果 } rank[j] \leqslant w \\ 0, \text{否则} \end{cases}$$

则 ξ_{ij} 为路径 (i,j) 上的信息权重。应用信息权重 ξ_{ij},蚂蚁可由

城市 i 按下式的概率选择城市 j：

$$P_{ij}^{k}(t)=\begin{cases}\dfrac{\xi_{ij}\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{r\in allowed_k}\xi_{rj}\tau_{ir}^{\alpha}(t)\eta_{ir}^{\beta}(t)}, & j\in allowed_k\\ 0, & \text{否则}\end{cases}$$

上式中，对各路径上的信息量及期望程度都乘上了信息权重 ξ_{ij}。路径 (i,j) 的信息权重 ξ_{ij} 反映了蚂蚁从城市 i 选择下一城市 j 时，路径 (i,j) 上的信息量 $\tau_{ij}(t)$ 以及可访问度 $\eta_{ij}(t)$ 对蚂蚁选择概率的影响程度。对于信息量较大的路径，其信息权重较大，蚂蚁选取该路径的概率也较大。

由上面的论述可知，各路径上的信息权重是公比为 $q(q\leqslant 1)$ 的等比数列。而当上一次迭代中蚂蚁遍历经过城市 i 为起点的路径较集中时，城市 i 的聚度就较大，这就使得分布范围 w 以及 q 值较大，因而各条路径的信息权重 ξ_{ij} 的差距就比较小。这样，在下次迭代中蚂蚁遍历经过城市 i 为起点的路径被选择的概率分布就比较均匀。特别地，当 $w=r$ 时，$q=1$，所有路径的信息权重的值均为 1，这就达到了完全平均。相反地，当上一次迭代中蚂蚁遍历经过城市 i 为起点的路径较分散时，城市 i 的聚度就较小，使得分布范围 w 以及 q 值较小，因而各条路径的信息权重 ξ_{ij} 差距就比较大，以城市 i 为起点的路径被选择的概率分布就比较集中。特别当 $w=1$ 时，仅有信息量最大的一条路径上的信息权重为 $1/r$，其余所有路径的信息权重的值均为 0。此时，蚁群就会仅选择信息量最大的那条路径。该算法的框架描述如下：

算法 1. 自适应蚁群算法

(1) 初始化

随机产生 m 个初始解，设其中经过路径 (i,j) 的初始解有 s 个，它们的总长度分别为 $L^1,L^2,\cdots,L^s$，则路径 (i,j) 上的初始信息量为：

$$\tau_{ij}(0) = \sum_{k=1}^{s} Q/L^k$$

式中，Q 为常数。

(2) 迭代过程

while 不满足结束条件 do

(2.1) for $i=1$ to n do (对 n 个城市循环)

计算城市 i 的聚度及本次迭代蚂蚁的分布范围 w

for $k=1$ to m do (对 m 个蚂蚁循环)

(2.1.1) 根据概率选择算法 2　选择下一城市 j

(2.1.2) 局部更新路径(i,j)上的信息量

end for k

m 只蚂蚁中若有当前已遍历的路径长度和已经超过上一次迭代得到的最优路径长度的，则终止该蚂蚁本次迭代的遍历

end for i

(2.2) 对所有城市的各条路径整体更新其信息量

end while

算法 2. 选择算法

{

比较 w 与 $allowed_i$ 中的城市数目 num：

若 $num < w$，

$$j = \begin{cases} \text{argmax}\tau_{ij}(t)\,(j \in allowed_i), & q \leqslant q_0 \\ \text{按概率 } P_{ij}^k \text{ 选取城市 } j, \text{否则} \end{cases}$$

因为 $num < w$，此时可能是：

(1) w 比较大，即聚度较大，从而要求本次迭代中蚂蚁对路径的选择概率分布均匀；

(2) num 比较小，蚂蚁本次的搜索已接近尾声。

因此，可参考选择概率自适应的方法，通过设定一个阈值 $0.4r$ 来限定 q_0 的取值，从而确定最优信息被强化的程度。当 $w < 0.4r$

时,即(2)的因素多些,则取 $q_0 = 0.8$,以较大的概率选择最优路径,使蚂蚁趋向于集中到最优路径上;当 $w \geq 0.4r$ 时,即(1)的因素多些,则取 $q_0 = 0.2$,使蚂蚁对路径的选择比较均匀。

若 $num >= w$,

将 $allowed_i$ 集合中各条路径上的信息量由大到小排序,选择前 w 条路径,并将其序号记入数组 $rank$ 中。计算与城市 i 相连的各路径的信息权重,并选择城市 j。

}

4.4.4.2 自适应的信息量更新策略

一些典型算法在更新信息量的时候,要么对所有的蚂蚁所经过的路径增加其信息量,要么只增加最优适应度的路径上信息量,其余信息量被削减;或者就是基于固定等级的算法,让适应度相对较好的若干条路径,根据其解的优劣程度决定信息量的增加程度。这些做法都采用了固定的信息量增减的比例,而忽视了解的分布情况。按照大量学者在研究中的实验经验,应根据信息量的均匀度自适应地进行信息量的更新,以动态地调整各路径上的信息量的分布,使其不至于过分集中或者分散,以在加速收敛的同时避免早熟。算法 1 中第 2.1.2 行的信息量局部更新可根据以下策略:

$$\tau_{ij}(t-1) = \begin{cases} \tau_{ij}(t) - 10/d_{ij}, & \text{若本次迭代中已有 } m/3 \text{ 只蚂蚁选择同一路径}(i,j), \text{或 } m/5 \text{ 只蚂蚁选择该路径后终止本次迭代的遍历} \\ \tau_{ij}(t) + 1/d_{ij}, & \text{否则} \end{cases}$$

由于蚂蚁常常选择信息量较大的路径,当多只蚂蚁选中同一路径后,信息量增加的幅度太大就容易使多只蚂蚁集中到该路径,所以我们取 $1/d_{ij}$ 为增加的信息量。若选择该路径的蚂蚁达到一定数量,或多数蚂蚁选择该路径后因当前距离超过上一次的最优路径长度而终止遍历(这里分别设定为 $m/3$ 和 $m/5$),信息量可取为 $-10/d_{ij}$。这时可大幅度削减其信息量,使其趋于各条路径信息

量的平均值，从而使蚂蚁选择其他路径的可能性增加，让搜索得到的解趋于多样化。

算法 1 中的第 2.2 行可按下式进行信息量的整体更新：

$$\tau_{ij}(i+1) = (1-\rho)\tau_{ij}(t) + \sum_{l=1}^{m} \psi_l \bar{A}\tau_{ij}^{l}(t)$$

式中，

$$\bar{A}\tau_{ij}^{l}(t) = \begin{cases} Q/L^{l}(t), & \text{第 } l \text{ 只蚂蚁经过路径}(i,j) \\ 0, & \text{否则} \end{cases}$$

$L^{l}(t)$ 为本次迭代中第 l 只蚂蚁遍历的路径全长，ψ_l 为第 l 只蚂蚁所对应的解对该路径上信息量更新的影响程度。ψ_l 的计算方法如下：设经过路径(i,j)的蚂蚁总数为 k，对它们在本次迭代中遍历的路径全长由小到大进行排序，所得序号存放于数组 $rank_1$ 中，即 $rank_1[l]$ 表示第 l 只蚂蚁对应的序号。我们从以下三个方面来说明决定 ψ_l 的原则性要求：

① 从由城市 i 出发的各条路径来看。路径(i,j)的信息权重较高，该路径上的信息量也就较高，此时$(1-\xi_{ij})k$ 的值就相应较小。对于较少的优秀解，由于它们的 $rank_1[l]$值比较小，ψ_l 应为正值，在该路径上将增加信息量；而对较多的其他解，由于它们的 $rank_1[l]$值比较大，它们的 ψ_l 为负值，在该路径上将减少信息量，这样会使路径(i,j)上的总体信息量减少。反之，当路径(i,j)的信息权重比较低时，会使该路径上的总体信息量增加，这样可以调节各路径上的信息量，防止过于集中。

② 从各个蚂蚁本次所经过的路径来看。若第 l 只蚂蚁遍历时路径总长度比较小，即 $rank_1[l]$值比较小，则 ψ_l 应为正值，路径上的信息量强度增加；而且 $rank_1[l]$越小，即长度越短，路径上信息量增加的程度越大，这样就有效地强化了短路径上的信息。反之，若第 l 只蚂蚁遍历时路径总长度比较大，则将减少相应路径上的信息量强度；而且 $rank_1[l]$越大，即长度越长，路径上信息量减

少的程度越大。这样可在防止信息量过分集中的同时保持较好路径上的信息量。

③ 从城市 i 本身的聚度来看。当城市 i 的聚度比较高时，与它相连的各条路径的信息权重 ξ_{ij} 差距就比较小，ψ_l 的差异也比较小，各个解对信息量的影响就比较均匀，以避免较优解对信息量的更新有较大的影响。反之，当城市 i 的聚度比较低时，ψ_l 的差异就应比较大，较优解会对信息量的更新有较大的影响，以使较优解的路径上集中较高的信息量。因此，这种自适应的信息量更新机制可以动态调节信息量，在蚂蚁的搜索速度和解的多样性之间取得较好的平衡。

4.4.5 动态蚁群算法

动态蚁群算法相对于传统蚁群算法和最大最小蚁群算法，其主要的改进在于以下几点：传统蚁群算法和最大最小蚁群算法依据信息素和启发函数来选择目标城市，如何依据这两个因子选择城市对算法的性能是非常关键的。而动态蚁群算法在迭代过程中，在选择目标城市的标准时，没有使用一个固定的标准，这样就有利于减少进化停滞现象。在传统蚁群算法中，对信息素的强度没有限制，因而易于陷入局部最优点。而最大最小蚁群算法对信息素的强度给予了一定的范围限制，从而大大改善了算法的性能。动态蚁群算法中，挥发因子是动态变化的。信息素浓度越高，则挥发因子越大；浓度越低，则挥发因子越小。这样实际上就对信息素的浓度也进行了限制，使信息素不可能无限增大，也不可能为零。最大最小蚁群算法中，仅有最好蚂蚁所走路径上的信息素会进行全局更新。如果能对更多路径上的信息素进行更新，则有可能加快演化的速度。

4.4.5.1 权函数

传统蚁群算法和最大最小蚁群算法依据 $[\tau(r,s)]^{\alpha} \cdot [\eta(r,s)]^{\beta}$ 来选择下一个转移城市，其中 α,β 是常数。在自然界中，生

物随着时间的推移对环境适应之后，不再对环境产生敏感。基于此原理，可认为蚂蚁随着时间的推移对信息素慢慢变得不敏感。在算法中可体现为：$[\tau(r,s)]^{\alpha}$ 随着时间的进行相对于 $[\eta(r,s)]^{\beta}$ 逐渐减小，这样，信息素的影响减小了，而启发函数的影响则相对增加，这样 α,β 就变成与时间有关的函数。在 TSP 问题中，$\eta(r,s)=1/d_{rs}$，d_{rs} 是城市 r 和 s 之间的距离。为简化计算，可置 $\alpha=1$，而 β 按迭代次数进行如下变化：

$1 \sim mn/3$ 次迭代间 $\beta=5$；

$mn/3 \sim mn/2$ 次迭代间 $\beta=4$；

$mn/2 \sim mn \cdot 2/3$ 次迭代间 $\beta=3$；

$mn \cdot 2/3 \sim mn$ 次迭代间 $\beta=2$。

mn 是总的迭代次数。循环次数可以在仿真中加以确定。

4.4.5.2 *动态挥发因子*

在传统蚁群算法中，挥发因子是一个常数。而在真实世界中，信息素浓度越高，会导致挥发越快；信息素浓度越低，则挥发越慢。这样就可以防止信息浓度无限制地增长，或者很快变为零，导致陷入局部最优。在这种变化模式下，挥发因子由常数变成了以 $\tau(r,s)$ 为变量的函数。

这时，局部信息素更新规则变为：

$$\tau(r,s) \leftarrow [1-\rho(\tau(r,s))]\tau(r,s) + coe[\tau(r,s)]\Delta\tau$$

挥发函数对算法的性能有直接的影响，事实上最大最小蚁群算法可以看作动态蚁群算法的一个特例。

4.4.5.3 *最优、最差路径信息素全局更新*

传统蚁群算法和最大最小蚁群算法中，仅对最好路径上的信息素进行全局更新，即仅有一只蚂蚁对全局信息素的更新产生影响。如果有多只蚂蚁对全局信息素的更新产生影响，则可加速演化过程。更新规则可设计如下：

$$\tau(r,s) \leftarrow [1-a(\tau(r,s))]\tau(r,s) + coe(r,s)\Delta\tau(r,s)$$

以上规则可以实现对两只蚂蚁所走路上的信息素进行更新，即对最好路径及最差路径上的信素进行全局更新，参数：

$$coe(r,s) = \begin{cases} C, (r,s) \in global\_best\_tour \\ -C, (r,s) \in global\_worst\_tour \\ 0, (r,s) \in the\_other \end{cases}$$

公式变为：

$$\tau(r,s) \leftarrow [1 - a(\tau(r,s))]\tau(r,s) + C\Delta\tau(r,s): best$$

$$\tau(r,s) \leftarrow [1 - a(\tau(r,s))]\tau(r,s) - C\Delta\tau(r,s): worst$$

式中，

$$\Delta\tau(r,s) \doteq \begin{cases} (L_{gb})^{-1}, (r,s) \in global\_best\_tour \\ (L_{gw})^{-1}, (r,s) \in flobal\_worst\_tour \end{cases}$$

L_{gb}是蚂蚁所走的最好路径长度，L_{gw}是蚂蚁所走的最差路径长度。$a(\tau(r,s))$是全局信息素的挥发因子。

4.4.6 蚁群算法的并行实现

目前研究的各类蚁群算法主要是基于单 CPU 计算机系统实现的递推算法，其运行效率较低。而蚁群系统本质上是一个并行系统，应该采用并行化技术来提高运算速度。MPI 是一种用于并行编程的消息传递接口，具有可移植性和易用性等优点，提供了完备的异步通信功能以及进程组、集合通信、进程拓扑、通信上下文等高层概念。有方案在 MPI 的基础上，采用蚁群划分的策略实现了并行蚁群算法，并对该算法进行了仿真。所用实验环境为：四台 PⅢ 850，100MB 的交换式 Hub，网络节点数目 $N = 70$，蚁群规模 $M = 700$，迭代次数 $T = 1000$，每次运行时间为 308.85s（而单机运行时间为 1010.47s），加速比为 3.27，并行效率为 0.82。

并行蚁群算法的伪代码可描述如下：

```
begin
    系统初始化
        初始化 MPI 库，获取进程标号 P，组进程数目 Q；
```

初始化网络节点数目 N,蚂蚁数量 M,设置迭代次数 T;
初始化节点矩阵 $C[N][3]$,计算支路权值矩阵 $D[N][N]$;
设置蚁群系统基本参数 α,β,ρ,q_0,计算信息增量 τ_0;
设置各个进程蚂蚁队列中的蚂蚁起始位置;
设置最优路径 *best _ tour* = ∅,长度 *best _ len* = N * max ($D[i][j]$);
while (迭代次数 < T)
for ($k = P * M/Q$; $k < (P+1) * M/Q$; $k++$)
初始化蚂蚁 *Ants*[k]的起始位置和搜索禁忌表;
while(存在不属于搜索禁忌表中的节点)
if(随机变量 $q \leqslant q_0$)
按照 $\arg\max_{u \in J_k(r)}[\tau(r,u)] \cdot [\eta(r,u)]^{\beta}$
选择节点 s_k;
else
以转移概率 $P_k(r,s)$随机性选择节点 s_k;
end if
蚂蚁 *Ants*[k]移动到节点 s_k,修改搜索禁忌表;
应用局部更新规则,修改最新支路上的信息量;
end while
蚂蚁 *Ants*[k]移动到起始节点;
应用局部更新规则,修改该支路上的信息量;
end for
检索本次循环中找到的最优路径,比较并修改全局最优路径;
应用全局更新规则修改当前最优路径中各个支路上的信息量;
MPI _ Barrier( )使用路障机制同步集合中的所有进程;

MPI _ Reduce( ) 使用规约操作收集各进程的支路信息痕迹;

采用加权平均的方法计算总的支路信息痕迹;

MPI _ Bcast( ) 使用广播操作统一各进程中的支路信息痕迹;

end while

MPI _ Reduce( ) 使用规约操作获取当前进程集合搜索到最短路径;

由搜索到最短路径的进程输出全局最优路径和最优路径长度;

end

从上述算法可以看出,蚁群算法具有良好的计算格式,只要能够确定网络的节点集合和支路权值矩阵,就可以采用该算法进行优化。各节点参数可采用三维坐标表示,支路权值矩阵由 CAD 软件生成,而上述算法中的权值由两个节点间的距离或连接线的长度表示。

4.4.7 具有感觉和知觉特征的蚁群算法

科学研究表明,蚂蚁对于外界刺激物有着惊人的感觉和知觉能力。在蚁群中,单只蚂蚁的能力和智力非常有限,但是整个蚁群可以有足够的能力来完成筑巢、觅食、迁徙、清扫巢穴等复杂行为。它们可以在各自的感觉和知觉能力支配下,很好地通过相互协调、分工、合作而完成任务。因此,我们可以在蚁群算法中模拟蚂蚁的这种感知现象,让蚂蚁根据知觉和感觉规律选择路径,以在加快收敛速度的同时保证解的多样性。根据感知规律,可以将蚁群算法中蚂蚁的搜索过程分为以下三个阶段。

4.4.7.1 *蚂蚁搜索的初始阶段*

心理学研究指出,感觉和知觉是客观事物作用于神经系统,通过引起神经系统的活动而产生的。产生感觉和知觉的神经机构叫

分析器,感受性是分析器对相应刺激的感觉能力。感觉是大脑对当前直接作用于感觉器官的客观事物个别属性的反映,而知觉是其整体属性的反映,两者是紧密相关的。但在实际的生物系统中,并不是所有的刺激都能引起人或者动物的感觉,只有达到一定量的刺激才能引起相应的感觉。现实生活中,往往会出现这样的情况:同是一种刺激,这个人感觉到了,另一个人却感觉不到,这就说明了他们的感受性是不同的。感受性是用感觉阈值的大小来度量的。感觉阈值是能引起感觉的、持续了一定时间的刺激量。心理学上把刚刚能引起感觉的最小刺激量,称为绝对感觉阈值(Absolute Sensor Threshold)。

在蚁群搜索过程的起始阶段,有的路径上有蚂蚁走过,有的路径还未来得及被涉足。而蚂蚁的路径选择策略是,一旦路径上有刺激物,即信息量多于其他路径,哪怕是很微弱的优势,它就会以较大的概率选择该路径,这就使得蚂蚁从搜索的一开始就以很大的概率集中到几条长度较短的路径上。在仿真实验中可以观察到,蚁群在搜索的初始阶段所得到的路径整体长度往往都偏大,导致了搜索所得的结果是局部最优而不是全局最优。

为了避免蚁群系统从搜索的一开始就失去解的多样性,受绝对感觉中阈值原理的启发,当路径上信息量的刺激量未达到蚂蚁的绝对感觉阈值时,可让蚂蚁忽视该刺激物的存在,也就是让蚂蚁在搜索初始阶段的路径选择不受信息量大小的影响。这里,我们可以将蚂蚁的绝对感觉阈值记为 AST。只有当信息量积累到超过 AST 时,蚂蚁才会在信息量的刺激下趋向于选择信息量较大的路径。通过这样的路径选择策略,就可以让蚂蚁在初始阶段选择较多的不同路径,以获得多样化的解,从而避免蚂蚁陷入局部最优,让蚂蚁尽量少走冤枉路。

4.4.7.2 蚂蚁搜索的中间阶段

刺激物引起感觉之后,如果刺激量发生了变化(增多或减

少),也会引起感觉的变化。但是,并不是刺激的所有变化量都能引起感觉。例如,如果在质量为100克的物体上只增加1克的质量,我们感受不到两者的差异。由于经验和潜意识的作用,当刺激条件的改变幅度仅局限于一定的范围内时,包括特定感觉的知觉的映象仍然会保持相对不变,这就是知觉的"恒常性"。只有当刺激变化到一定量时,才能使我们感觉到差别。能引起差别感觉的刺激物的最小变化量,称为差别感觉阈值(Contrast Sensor Threshold,CST)。早在19世纪前半期,德国心理学家韦伯在研究差别感觉阈值时发现,差别感觉阈值是随原来刺激量的变化而变化的,而且表现了一定的规律性。在一定的范围内,差别感觉阈值与原来量的比值是一个常数。如果将它用公式来表示,设 I 表示原来刺激物的强度,ΔI 表示差别感觉阈值,那么在 I 小于某个特定的限度 IT(Intensity-Threshold)时,就有:

$$\Delta I/I = K$$

这就是著名的韦伯定律(Weber's Law)。此处 K 是一个常数,称为韦伯系数。质量感觉的 K 为1/30,听觉的 K 为1/10,视觉的 K 为1/100。

由于各条路径上的信息量也是在不断变化的,我们可将上述规律应用到蚁群算法的搜索过程中。这里,我们改变传统蚁群算法中信息量一旦变化就直接影响蚁群路径选择的做法,认为蚁群在一定的刺激强度范围内也存在一个差别感觉阈值。当路径上信息量的增加或减少的量在差别感觉阈值之下时,蚂蚁就遵循知觉的恒常性规律,感受不到该路径上信息量的变化。这样,该条路径被选择的概率主要依据于以前迭代中蚂蚁的路径选择经验形成的潜意识作用;反之,蚂蚁就受其显意识控制,按照路径上所有蚂蚁信息量的高低决定路径选择概率的大小。根据韦伯定律,在具体实现此过程时,可以取:

$$CST = \tau_{ij}(t)K$$

式中，K 为蚂蚁对信息量感觉的韦伯系数，本文取值为 1/50。若记：

$$\theta_{ij}^{k}=\begin{cases}\dfrac{\alpha_{ij}^{k}}{\beta\ k_{ij}}, \Delta\tau_{ij}\leqslant CST\\ \tau_{ij}(t)\eta_{ij}, \text{否则}\end{cases}$$

$$\text{式中：}\alpha_{ij}^{k}=\frac{\tau_{ij}^{k}(t)}{\sum\limits_{h\in allowed_k}\tau_{ih}^{k}(t)}\ ,\ \beta_{ij}^{k}=\frac{\tau_{ij}(t)-\tau_{ij}^{k}(t)}{\sum\limits_{h\in allowed_k}[(\tau_{ih}(t)-\tau_{ij}^{k}(t)]}$$

$\tau_{ij}^{k}(t)$ 表示至 t 时刻蚂蚁 k 在路径 (i,j) 上历次遗留的总信息量，$\eta_{ij}=1/d_{ij}$。这里，我们用 α_{ij} 表示在蚂蚁的潜意识作用下路径 (i,j) 对它的吸引程度，β_{ij} 表示该路径对其他蚂蚁的吸引作用。实际上，当蚂蚁 k 按照其潜意识选择路径时，β_{ij} 包含了所有其他蚂蚁所留信息量的作用，会干扰蚂蚁 k 的潜意识，构成潜意识作用下蚂蚁 k 对路径 (i,j) 的排斥作用。这样，蚂蚁 k 选择路径 (i,j) 的概率表示为：

$$P_{ij}^{k}=\begin{cases}\dfrac{\theta_{ij}^{k}}{\sum\limits_{h\in allowed_k}\theta_{ij}^{k}}, j\in allowed_k\\ 0, \text{否则}\end{cases}$$

当路径 (i,j) 上的信息量变化 $\Delta\tau_{ij}$ 小于绝对感觉阈值时，蚂蚁按照自己的潜意识作用选路。此时，如果蚂蚁 k 在某一条路径上走过的次数越多，它对这条路径就越熟悉，其潜意识作用下选择该路径的概率就大；反之，当一条路径上其他蚂蚁的信息量越多，则潜意识中蚂蚁 k 对该路径就越排斥，该路径被选择的概率就小。这种机制有效地防止了蚂蚁一味相信他者而造成的盲从现象，大大降低了大量蚂蚁聚集于少数局部较优路径上造成早熟、停滞现象的可能性。当信息量变化较大时，蚂蚁在显意识作用下受整体信息量的影响而进行路径选择，遵循路径上走过的蚂蚁越多，选择该路径的概率越大的原则，趋向于选择优势较大的路径（这里的

优势主要指其信息量较大和长度较短),保证了所选路径的优越性,并且加快了收敛速度。

4.4.7.3 蚂蚁搜索的结束阶段

由于韦伯定律只能在刺激物的强度 I 小于某个特定的限度 IT 时才能适用,在超过 IT 时并不适用,因此,当路径上信息量达到相当大的程度时,我们应改变蚂蚁的路径选择策略。为了确定 IT 的值,首先应估计各个路径上可能的最大信息量 τ_{max}。由于一条路径上的最大信息量只能出现在所有蚂蚁的每次迭代都选择该路径的情况下,若预定蚂蚁搜索的最大遍历次数为 NC,蚁群总数为 m,则此时的最大信息量可以表示为:

$$\tau_{max} = NC \cdot m \cdot \text{适应度的数量级}$$

从绝对感觉阈值的定义可知,此处适应度的数量级应该与其相当,因此,可以取:

$$IT = h \cdot \tau_{max} = h \cdot NC \cdot m \cdot AST$$

式中,h 为一常数,可取 1/2、2/3 或 3/4 等数值。应该看到,某路径上信息量强度较大,可能是因为进入了局部最优的状态,或者是因为整个蚁群的遍历行为已经接近尾声。但由于上述搜索中间阶段的选路策略已经有效地避免了停滞现象的产生,所以此时不大可能是因为陷入局部最优,而是因为这些路径占有的绝对优势。为加快收敛,此时可按照类似于 Ant-Q 中的选路策略来选择下一城市 j:

$$j = \begin{cases} \text{argmax}\{\tau_{ij}(t) \mid j \in allowed_k\}, q \leqslant q_0 \\ \text{蚂蚁 } k \text{ 以下面 } P_{ij}^k(t) \text{ 的概率选择城市 } j, \text{否则} \end{cases}$$

$$P_{ij}^k(t) = \begin{cases} \dfrac{\tau_{ij}(t)\eta_{ij}(t)}{\sum_{allowed_k}\tau_{ij}(t)\eta_{ij}(t)}, j \in allowed_k \\ 0, \text{否则} \end{cases}$$

式中,$\eta_{ij}(t) = 1/d_{ij}$,q 为产生的随机数,q_0 为一常数。

4.4.7.4 算法框架

综上所述,具有感觉和知觉特征的蚁群算法的框架可以描述如下:

```
main(  )
{
```

(a) 初始化

随机产生 m 个初始解,计算这 m 个初始解的适应度,即 m 只蚂蚁遍历完成的路径长度的倒数,记其最大适应度为 f_{max}。取 $AST = Cf_{max}$,式中,C 为常数,我们取其值为 5。设经过路径(i,j)的初始解有 m 个,它们的总长度分别为 $L_1, L_2, \cdots, L_s$,则路径(i,j)上的初始信息量为:

$$\tau_{ij}(0) = \sum_{k=1}^{s} 1/L_k$$

(b) 迭代过程

```
while not 结束条件 do
for i = 1 to n do (对 n 个城市循环)
{
for k = 1 to m do (对 m 个蚂蚁循环)
    if (以城市 i 为起点的各条路径的平均信息量未达到 AST)
    then
    {
        1.1 蚂蚁在当前城市随机产生下一次访问的城市序号;
        1.2 更新蚂蚁 k 的信息量;
    }
    else
    {
        2.1 if(以城市 i 为起点各条路径的平均信息量小于 IT)
        then
```

根据概率 P_{ij}^{k} 选择下一城市 j;

else 根据类似于 Ant-Q 中的选路策略选择下一城市 j;

2.2 局部更新(i,j)上的信息量;

}

end for i

3.1 对所有蚂蚁的路径整体更新其信息量;

end while

}

4.4.7.5 自适应的信息量更新策略

一些典型的蚁群算法在更新信息量时,要么采取只要蚂蚁遍历时经过该条路径就增加其信息量的方法;要么只让最优适应度路径上的信息量得以增加,其余路径上的信息量被削减;要么就是基于等级的算法,让适应度相对较好的固定若干条路径根据其解的优劣程度决定信息量的增加幅度。这些做法都采用了固定的信息量增减的比例,忽视了解的分布情况。为此,这里的自适应更新策略中,可根据信息量的分布情况进行信息量的更新,以动态地调整各路径上的信息量的分布,使之不至于过分集中或者分散,以在加速收敛的同时避免早熟。信息量的局部更新可根据以下策略:

$$\tau_{ij}(t+1)=\begin{cases}\tau_{ij}(t)-5/d_{ij}, & \text{如果在此之前已有 } m/2 \text{ 只蚂蚁选择了同一城市 } j, \text{或有 } m/4 \text{ 只蚂蚁后选择该城市终止本次遍历}\\ \tau_{ij}(t)+1/d_{ij}, & \text{否则}\end{cases}$$

这里,可认为在一次迭代中,蚂蚁在一个城市选择下一城市的时间间隔内信息量尚未被挥发,所以,局部更新时就不考虑蒸发因子。由于蚂蚁常常选择信息量较大的路径,当多只蚂蚁选中同一路径后,信息量增加的幅度太大,就容易使多只蚂蚁集中到该路径,因此只取 $1/d_i$ 为增加的信息量;若选择该路径的蚂蚁达到

$m/2$，或有 $m/4$ 的蚂蚁选择该路径后因当前距离超过上一次的最优路径长度而终止遍历，则减少 $5/d_{ij}$ 的信息量，以使其趋于各条路径信息量的平均值，从而使蚂蚁选择其他路径的可能性增加，让搜索得到的解趋于多样化。

上述算法的 1.2 和 3.1 行分别按下面的两式整体更新信息量：

$$\tau_{ij}^{k}(t+1) = (1-\rho)\tau_{ij}^{k}(t) + \psi_k \Delta\tau_{ij}^{k}(t)$$

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{k=1}^{m} \psi_k \Delta\tau_{ij}^{k}(t)$$

式中，$\Delta\tau_{ij}^{k}$ 为蚂蚁 k 在本次循环中在路径 (i,j) 留下的信息量，$L_k(t)$ 为本次迭代中第 k 只蚂蚁遍历的路径全长，ψ_k 为第 k 只蚂蚁所对应的解对该路径上信息量更新的影响程度。ψ_k 的计算方法如下：首先，对经过路径 (i,j) 的总数为 s 的蚂蚁在本次迭代中遍历的路径全长由小到大进行排序，所得序号存于数组 $rank$ 中，$rank(k)$ 表示第 k 只蚂蚁对应的序号，取：

$$\psi_k = s/2 - rank[k] + 1$$

上式中，若 $rank[k]$ 值比较大，即第 k 只蚂蚁遍历时路径总长度较长，如果超过了其他一半以上蚂蚁的路径长度，那么 ψ_k 就为负值，整体更新时该蚂蚁就减小相应路径上的信息量强度。而 $rank[k]$ 越大，即长度越长，路径上信息量减小的程度越大，从而使得越差路径上的信息量保留得越少，有利于保持较好路径上信息；反之，若 $rank[k]$ 值较小，ψ_k 就为正值，则蚂蚁在 (i,j) 上增强信息量，且 $rank[k]$ 越小，即路径越短，则 ψ_k 越大，信息量增加的强度就越大，这样就有效地强化了短路径上的信息。这种自适应的信息量更新机制可以动态地调整信息量，有效地实现蚂蚁的搜索速度和解的多样性之间的平衡。

4.5 广义蚁群算法及收敛性分析

4.5.1 广义蚁群算法的基本步骤

本节将在蚁群算法基本思想的基础上描述一种通用的优化算法——广义蚁群算法，并在压缩映象的不动点理论基础上对其收敛性进行讨论。

对于一般定义在紧集合上的约束优化问题，可用外点法构造辅助函数，将不便于计入可行域的约束以罚函数计入目标函数中，其表达式为：

$$\min F(X) = \min\{f(X) + \sigma_1 \sum_{i=1}^{l_0} [h_i(X)]^2 + \sigma_2 \sum_{j=1}^{u_0} [\max(0, -g_j(X))]^2\}$$

$$h_i(X) = 0, i = 1,2,\cdots,l - l_0$$

$$g_j(X) \geqslant 0, j = 1,2,\cdots,u - u_0$$

式中，$X = (x_1, x_2, \cdots, x_n)^T$ 为待优化矢量；l, u 分别为原问题等式约束和不等式约束的个数。

上式的可行域可记为：

$$S = \{X \mid h_i(X) = 0, i = 1,2,\cdots,l - l_0,\ g_j(X) \geqslant 0,$$
$$j = 1,2,\cdots,u - u_0\}$$

为了加速迭代，本文中 σ_i 与当前迭代次数 k 有关：

$$\sigma_i(k) = [(2/(1 + e^{-\frac{\alpha k}{T}})) - 1]\sigma_i^0, i = 1,2$$

式中，α 为正系数，用以调节 σ_i 的变化速度；σ_i^0 为 $\sigma_i(k)$ 上限；T 为迭代次数上限。

上式使 $\sigma_i(k)$ 由0逐步趋近于 σ_i^0。开始时，惩罚系数较小，有助于大范围搜索，后来逐步变大，使最终结果即为原问题所求。

经过处理，我们所描述的广义蚁群算法的基本运算步骤为：

(1) 初始化

当前迭代次数 $k = 0$，在 S 中取 N 组服从均匀分布的随机数，

给定 N 只蚂蚁的初始位置，形成初始蚁群 C^0，且：

$C^0=(X_1,X_2,\cdots,X_N)^{\mathrm{T}}$，其中，$X_i\in S$。

初始化信息密度矩阵 $\tau_{N\times N}$，$\tau_{ij}=\tau_0$，τ_0 为一小正数。

设 $b(i)$ 为位置 i 的蚂蚁数目，初始时：$b(i)=1$，$i=1,2,\cdots,N$，令初始化蚂蚁可见域为 $D(k)$，当：

$$\|X_i-X_j\|\leqslant\tau_{ij}D(k)$$

蚂蚁 i，j 可互相移动至对方位置。$\|\cdot\|$ 为某种范数。为减小计算量，本文采用 $\|\cdot\|_2$，即为 R^n 上的平方范数。$D(k)$ 与迭代次数 k 有关，这里采用：

$$D(k)=2\{1-[1/(1+\mathrm{e}^{\frac{ak}{T}})]\}D_0$$

式中，D_0 为可见域上限。

上式使每只蚂蚁开始能进行大范围搜索，后来再逐步精细化。

(2) 搜索策略

对于蚂蚁位置 $i(i=1,2,\cdots,N)$，当 $b(i)\geqslant 1$ 时，形成集合 A_i，且：

$$A_i=\{X_j\mid\|X_i-X_j\|\leqslant\tau_{ij}D(k)\}$$

当 $A_i\neq\varnothing$ 时，转步骤(3)；当 $A_i=\varnothing$ 时，转步骤(4)。其中 $\varnothing$ 为空集。

(3) 计算

设 A_i 中元素个数为 m，计算：

$$\eta_{ij}=F(X_i)-F(X_j),X_j\in A_i$$

$$l=\frac{1}{m}\{[2/(1+\mathrm{e}^{\frac{ak}{T}})]-1\}\sum_{X_j\in A_i}\eta_{ij}$$

令 $\eta_{ij}^{\min}=|\min(\eta_{ij})|$，$G=(1+\varepsilon)\eta_{ij}^{\min}$，$\varepsilon$ 为一小正数。

定义转移概率：

$$P_0=\frac{(1+G)^{\gamma_1}\left(\frac{1}{m}\sum_{X_j\in A_i}\tau_{ij}\right)^{\gamma_2}}{\sum_{X_j\in A_i}(\eta_{ij}+G)^{\gamma_1}(\tau_{ij})^{\gamma_2}+(1+G)^{\gamma_1}\left(\frac{1}{m}\sum_{X_j\in A_i}\tau_{ij}\right)^{\gamma_2}}$$

$$P_{ij} = \frac{(\eta_{ij} + G)^{\gamma_1}(\tau_{ij})^{\gamma_2}}{\sum_{X_j \in A_i}(\eta_{ij} + G)^{\gamma_1}(\tau_{ij})^{\gamma_2} + (1 + G)^{\gamma_1}\left(\frac{1}{m}\sum_{X_j \in A_i}\tau_{ij}\right)^{\gamma_2}}$$

式中，γ_1，γ_2 为常数，取 $\gamma_1 = \gamma_2 = 1$。

由上两式可得：

$$P_0 + \sum_{X_j \in A_i} P_{ij} = 1$$

随着 $F(X_j)$ 的减小，τ_{ij} 增大，P_{ij} 增大。P_0 为进行邻域搜索的概率，与可见域中蚂蚁对应目标函数的平均值及当前迭代次数有关。

对 P_0，P_{ij} 进行赌轮选择，以选择不同的修正规则。

若选中某一 P_{ij}，则执行修正规则 1：

修正规则 1 X_i 处蚂蚁转向 X_j 处，$\Delta\tau_{ij} = P_{ij}$；$b_i = b_i - 1$，$b_j = b_j + 1$；$X_i \leftarrow X_j$。转向步骤(5)。

若选中 P_0，则执行修正规则 2：

修正规则 2 在 X_i 邻域中用某种优化算法进行搜索，邻域定义为：

$$S_{X_i} = \{Y \mid [\ \| X_i - Y \| \leqslant \beta D(k)\ ]I \quad S\}$$

式中，$\beta \in (0,1)$ 为系数。

设搜索结果为 Y，则：

$$X_i \leftarrow Y$$

$$\Delta\tau_{ij} = \frac{\{[F(X_i) - F(Y)] + G\}^{\gamma_1}\left(\frac{1}{m}\sum_{X_j \in A_i}\tau_{ij}\right)^{\gamma_2}}{\sum_{X_j \in A_i}(\eta_{ij} + G)^{\gamma_1}(\tau_{ij})^{\gamma_2}}$$

转向步骤(5)。

(4) 直接邻域搜索

搜索域同规则 2。如搜索结果为 Y，则执行修正规则 3。

修正规则 3

$$X_i \leftarrow Y, \Delta\tau_{ij} = r \quad X_j \in A_i$$

式中，r 为常数。

(5) 修正信息密度矩阵

$$\tau_{ij}(k+1) = \rho\tau_{ij}(k) + \Delta\tau_{ij}$$

式中，ρ 为信息密度蒸发系数，$\rho \in (0,1)$。

注：①当 $\Delta\tau_{ij}$ 由修正规则 1 得到时，仅修正 i 至 j 的连接；当 $\Delta\tau_{ij}$ 由修正规则 2 得到时，对于所有 $X_j \in A_i$，且 $X_i \neq X_j$ 的 τ_{ij} 都修正；当 $\Delta\tau_{ij}$ 由修正规则 3 得到时，则所有 $X_j \in C^k$，且 $X_i \neq X_j$ 的 τ_{ij} 都修正。C^k 为第 k 次迭代形成的蚁群。②仅修正 τ_{ij}，τ_{ji} 不修正。

(6) 对所有 $X_i \in C^k$，都完成一次移动，统计结果为：①若不满足接受条件，取消本次迭代步骤(2)(3)(4)的结果，转步骤(2)。②若连续多次迭代蚁群统计结果不变，则对蚁群加扰动。可采取的扰动方法为扩大可见域、扩大邻域搜索范围：将任一只蚂蚁随机放置于 S 中某一点。多次扰动无效，输出结果。③当 $k < T$，$k = k+1$ 时，转步骤(2)；否则，输出结果。

4.5.2 广义蚁群算法收敛的充分条件

这里将基于压缩映象的不动点理论，对广义蚁群算法的收敛性进行简单的分析，并给出广义蚁群算法收敛的充分条件。

定义 1 广义蚁群算法空间 Ω：

$$\Omega = \{C^0, C^1, \cdots, C^k, \cdots\}$$

式中，$C^k = (X_1^k, X_2^k, \cdots, X_N^k)$ 为第 k 次迭代形成的蚁群，N 为蚁群规模(下同)，S 为可行域，$X_i^k \in S, i = 1,2,\cdots,N$。

由前面描述可知：$\forall X \in S$，都可找到 C^k，使 $X \in C^k$，且 $C^k \in \Omega$。

定义 2 Ω 中 C^i, C^j 距离 $d(C^i, C^j)$：

$$d(C^i, C^j) = \begin{cases} |M + F'(C^i)| + |M + F'(C^j)|, C^i \neq C^j \\ 0, C^i = C^j \end{cases}$$

式中，$F'(C^i)=\frac{1}{N}\sum_{i=1}^{N}F(X_i^i)$，$M=\varepsilon+|L|$，$X_i^i\in C^i$，$L$ 为 $F'(C^i)$ 的一个下界，由于待求为最小化问题，L 必存在；$C^i\in\Omega$；ε 为小正数。

结论1 (Ω,d) 构成完备的度量空间。

证明：首先，$d(C^i,C^j)$ 为 Ω 上的一个度量，

① $d(C^i,C^j)=d(C^j,C^i)$

② $d(C^i,C^k)+d(C^k,C^j)=|M+F'(C^i)|+|M+F'(C^k)|+|M+F'(C^k)|+|M+F'(C^j)|\geqslant|M+F'(C^i)|+|M+F'(C^j)|=d(C^i,C^j)$

③ $d(C^i,C^j)\geqslant 0$；$d(C^i,C^j)=0$，当且仅当 $C^i=C^j$

其次，Ω 是完备的：设 $\{C^i\}$ 是 Ω 中任一 Cauchy 列，则 $\forall\varepsilon>0$，存在 N，当 $i,j>N$ 时，有：

$$d(C^i,C^j)<\varepsilon$$

在上式中，令 $j\to\infty$，$C^i\to C(i\to\infty)$。由定义1可得 $C\in\Omega$。因此，(Ω,d) 是完备的度量空间。

由广义蚁群算法的描述可将其看成 $\Omega\to\Omega$ 的映射，记为 φ。则在完备的度量空间 (Ω,d) 上，φ 为 Ω 的自映象，当 φ 满足以下压缩映象条件之一时，φ 在 Ω 中存在唯一不动点。$\forall C^i,C^j\in\Omega$ 有：

$$d(\varphi C^i,\varphi C^j)\leqslant hd(C^i,C^j),h\in(0,1)$$

$$d(\varphi C^i,\varphi C^j)\leqslant h\{d(C^i,\varphi C^i)+d(C^j,\varphi C^j),h\in(0,1/2)\}$$

$$d(\varphi C^i,\varphi C^j)<\max\{d(C^i,\varphi C^i),d(C^j,\varphi C^j),d(C^i,C^j)\},\quad C^i\neq C^j$$

$$d(\varphi C^i,\varphi C^j)\leqslant h\max\{d(C^i,C^j),d(C^i,\varphi C^i),d(C^j,\varphi C^j),d(C^i,\varphi C^j),d(C^j,\varphi C^i)\},h\in(0,1)$$

满足上面四式的一个充分条件是：

$$F'(\varphi C^k)<F'(C^k)$$

因此,步骤(6)的①中的一个接受条件为:

$$F'(\varphi C^k) < F'(C^k)$$

由此可得以下的结论2:

结论2 在完备度量空间(Ω,d)中,若广义蚁群算法构成的映射$\varphi:\Omega\to\Omega$满足结论1中的条件,则φ存在唯一的不动点C^*,对任一$C^0\in\Omega$,蚁群算法形成的迭代序列:

$$\varphi^k C^0 \to C^*$$

式中,k为迭代次数。

第5章　基于蚁群算法的典型优化问题求解模式

典型的优化问题包括：旅行商问题、Flowshop 调度优化问题、约束优化问题、二次配置问题等。前面，作者已经将旅行商问题作为典型的组合优化类问题，举例讨论了该问题求解的基本蚁群算法及改进模式，并对几种改进的蚁群算法进行了详细讨论。这里，作者举一些其他的典型优化问题实例。

5.1　Flowshop 调度优化问题求解

作者以节点模式的 Flowshop 调度问题来说明蚁群算法的应用模式。

这里，我们首先将流水作业调度问题（Flowshop 问题）以节点或弧模式有向图表示，人工蚁受有向图上信息素踪迹的指引，在图

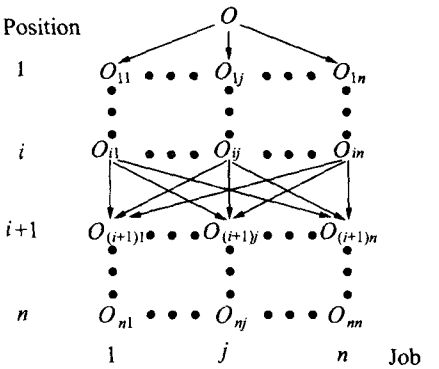


图 5－1　节点模式的 Flowshop 图

上进行搜索,并一步步地构造出问题的可行解。算法中的信息素踪迹更新过程作为蚁群间的间接通信机制,将引导整个蚁群收敛到问题的优化解。以下定义的信息素更新过程中的停滞状态脱离机制以及信息素限制机制能帮助人工蚁跳出局部最优解。算法局部搜索过程中所采用的基于关键路径的邻域结构缩小了问题的搜索空间。

高维的排列流水作业调度问题由于具有 NP 难度的计算复杂性,相关研究人员提出了许多启发式方法来加以求解。这些启发式方法大致可分为两类:构造型的基于问题的启发式方法(Constructive AdHoc Heuristic)和通用型的自然启发式方法(Meta-Heuristic),在前一类方法中,NEH 算法的总体性能最好;后一类方法有模拟退火(SA)、遗传算法(GA)、禁忌搜索(TS)、最大最小蚁群算法等。在一组 Taillard 标准测试问题集上,Fast-TS 算法效果最好,但该算法采用了 NEH 算法的结果作为初始解。

下面我们在 M. Dorigo 的蚁群算法(ACS)的基础上,给出一种新的蚁群优化算法,用于解决排列流水作业的优化调度问题。该算法模拟蚁群的自适应性觅食行为,通过一群人工蚁在人工信息素踪迹和基于问题的启发式信息的指引下,在问题空间移动,相互协作地求出空间中搜索问题的解。作为蚁群间的间接通信机制,算法中的信息素踪迹更新过程能引导整个蚁群收敛到问题的优化解。信息素踪迹更新过程中的停滞状态脱离机制以及信息素踪迹限制机制,可以帮助人工蚁跳出局部最优解。算法局部搜索过程中采用了基于关键路径的邻域结构,缩小了问题的搜索空间。

5.1.1 求解 Flowshop 调度问题的算法框架描述

5.1.1.1 解构造过程

流水作业调度问题的解由代表所有产品处理顺序的一个产品序列(路径)及其对应的完成时间表示。在此,我们提出本算法的两种解构造模式,这两种模式的主要区别在于其信息素浓度在不

同模式下有不同含义。

(1) 弧模式:类似于 TSP 问题,流水作业调度问题可用有向图的形式表示 $G=(N,A)$。其中,N 是图中节点集合,每个节点代表所要处理的作业,A 是图中弧的集合,每两个节点间由两条有向弧相连,表示作业间的先后顺序。

人工蚁在图 G 中搜索,通过遍历图中所有节点来构造问题的解。因而,Flowshop 调度问题可以转换为在有向图 $G=(N,A)$ 中寻找最佳路径的问题。每个人工蚁从一个虚构的起始节点(作业 0)出发,一步一步地构造出一条完全路径,在路径构造的每一步,蚂蚁 k 从节点 i 移动到相邻节点 s,节点 s 代表将被插入作业处理序列 $\pi'=(0,1,\cdots,i)$ 的下一个作业,s 的选择按照以下伪随机比例状态迁移规则进行:

$$s=\begin{cases}\underset{l\in N_i^k}{\operatorname{argmax}}[\tau_{il}(t)]^{\alpha}\cdot[\eta_{il}(t)]^{\beta}, & q<q_0\\ s', & q\geqslant q_0\end{cases}$$

上式中,随机数 $q\in[0,1]$,参数 $q_0\in[0,1]$ 决定了蚁群在图 G 中搜索时的知识利用与探索两者之间的权重差别。在搜索过程中,人工蚁以概率 q_0 移动到具有最大 $[\tau_{il}(t)]^{\alpha}\cdot[\eta_{il}(t)]^{\beta}$ 的值的邻域节点 $l\in N_i^k$,即人工蚁完全按照信息素及问题的启发式信息的指引来选择路径,这种情况我们称为知识的利用。与此对应,人工蚁以概率 $1-q_0$ 进行有偏向性的探索,在此情况下,当前位于节点 i 的人工蚁将以概率 $P_{is'}^k(t)$ 移动到邻域节点 s'。这里路径选择概率 $P_{ij}^k(t)$ 为:

$$P_{ij}^k(t)=\begin{cases}\dfrac{[\tau_{il}(t)]^{\alpha}\cdot[\eta_{ij}(t)]^{\beta}}{\sum\limits_{l\in N_i^k}[\tau_{il}(t)]^{\alpha}\cdot[\eta_{ij}(t)]^{\beta}}, & j\in N_i^k\\ 0, & j\notin N_i^k\end{cases}$$

式中,$\tau_{ij}(t)$ 是弧 $\mathrm{arc}(i,j)$ 上信息素的浓度,它代表作业 j 在作

业处理序列中对紧接作业 i 的期望程度，$\eta_{ij}(t)$ 是与弧 arc(i,j) 相关联的基于问题的启发式信息值，α,β 分别是控制信息素和启发式信息值在概率 $P_{ij}^{k}(t)$ 中权重的参数。本算法在状态迁移规则中未使用基于问题的启发式信息，因而我们令 $\alpha=1,\beta=0$。N_i^k 是当蚂蚁 k 位于节点 i 时的可行邻域，该邻域就是蚂蚁 k 还未访问的节点的集合，即蚂蚁 k 构造的部分解 π' 中未出现的节点集合。

(2) 节点模式：在解构造的节点模式下，代表 Flowshop 调度问题的有向图 $G=(N,A)$（见图 5-1）与在弧模式下图的表达形式有很大区别。在节点模式下，节点集合中 $N=\{O_{ij}\}$ 的节点 O_{ij} 代表作业 j 位于作业处理序列 π 的第 i 个位置。A 是图 G 中部分连接节点集 N 中各节点的有向弧的集合，连接节点 O_{ij} 和 $O_{(i+1),l}$ 的有向弧方向为从 O_{ij} 到 $O_{(i+1),l}$。这里的路径选择概率为：

$$P_{(i+1)j}^{k}(t)=\begin{cases}\dfrac{[\tau_{(i+1)j}(t)]^{\alpha}\cdot[\eta_{(i+1)j}(t)]^{\beta}}{\sum\limits_{l\mid O_{(i+1)l}\in N^k(O_{i\pi(i)})}([\tau_{il}(t)]^{\alpha}\cdot[\eta_{il}(t)]^{\beta})}, & O_{(i+1)j}\in N^k(O_{i\pi(i)})\\ 0, & O_{(i+1)j}\notin N^k(O_{i\pi(i)})\end{cases}$$

在节点模式下，上式中的 $N^k(O_{ij})=\{O_{(i+1),l}\mid l\notin\pi'\}$ 代表节点 O_{ij} 的可行邻域集合，其中 $l\notin\pi'$ 代表那些还未被人工蚁 k 调度的作业。此外 $\tau_{ij}(t)$，$\eta_{ij}(t)$ 分别表示对应于节点 O_{ij} 的信息素浓度和基于问题的启发式信息。信息素浓度 $\tau_{ij}(t)$ 意味着人工蚁在图 G 的搜索过程中从节点 $O_{(i-1)\pi(i-1)}$ 移到节点 O_{ij} 的渴望程度。其中 $P_{ij}^{k}(t)$ 表示人工蚁在探索过程中，从节点 $O_{(i-1)\pi(i-1)}$ 移动到节点 O_{ij} 的概率。

从虚构的起始点 job0 出发，人工蚁群运用下面的伪随机比例状态迁移规则，在图 G 中一步一步构造出问题的解，即：

$$\pi(i+1)=\begin{cases}\arg\max\limits_{l\mid O_{(i+1)l}\in Nk(O_{im(i)})}\{[\tau_{(i+1)l}(t)]^{\alpha}\cdot[\eta_{(i+1)l}(t)]^{\beta}\}, & \text{若 } q<q_0\\ S,\text{否则}\end{cases}$$

算法的其他方面与弧模式一致。

5.1.1.2 信息素更新模式

在算法的每一次迭代中，当整个蚁群中的所有人工蚁均已构造出问题的解时，图 G 上的信息素将被更新。信息素更新包含两个方面。首先，图 G 中的所有节点或连接节点的弧（依解构造过程的节点模式或弧模式而定）上的信息素将自然挥发掉一部分。此外，某些特殊的人工蚁将在其搜索路径上留下信息素踪迹。与传统蚁群算法不同（传统蚁群算法只允许找到目前最好解的人工蚁在其搜索路径上留下信息素），下面的算法在信息素更新过程中采用了一种蚁群种子策略，这种策略的核心是一个动态的蚁群种子集 θ。在算法的运行过程中，θ 的长度 λ 及其结构均是动态可变的，θ 由 λ 或 $\lambda-1$ 个全局精英人工蚁（即找到目前最好的前 λ 或 $\lambda-1$ 个解的人工蚁）和一个或零个局部精英人工蚁（即找到当前算法迭代过程中最好解的人工蚁）组成，θ 中的每个人工蚁所找到的解必须不相同。在 θ 中，全局精英人工蚁按照其解的质量以降序排列优先级。如果两个人工蚁解的质量相同，则新生成的人工蚁具有更高的优先级。如果 θ 包含局部精英人工蚁，其优先级可以任意给定或在算法运行过程中动态改变。算法中，所有在 θ 中的人工蚁将允许在其搜索路径中留下信息素。在解构造过程的两种不同模式（节点模式和弧模式）下，相应的信息素踪迹更新方式不同。在弧模式下，θ 中所有人工蚁在留下信息素时具有相同的优先级。而对于解构造的节点模式，θ 中具有高优先级的人工蚁将优先并独占式地在其搜索路径上留下其信息素，也就是说，如果 θ 中多个人工蚁的解包含相同的节点，则只有其中具有最高优

先级的人工蚁有权在这些节点上留下其信息素。

弧模式和节点模式的信息素踪迹更新规则可分别表示为：

弧模式：

$$\begin{cases}\tau_{ij}(t+1) = \max[\tau_{\min},(1-\rho)\cdot\tau_{ij}(t+1)+\rho\cdot f(\pi)], \\ \qquad ant(\pi)\in\theta \wedge \text{arc}(i,j)\in\pi \\ \tau_{ij}(t+1) = \max[\tau_{\min},(1-\rho)\cdot\tau_{ij}(t+1)], \\ \qquad ant(\pi)\notin\theta \vee \text{arc}(i,j)\notin\pi\end{cases}$$

节点模式：

$$\begin{cases}\tau_{ij}(t+1) = \max[\tau_{\min},(1-\rho)\cdot\tau_{ij}(t+1)+\rho\cdot f(\pi)], \\ \qquad ant(\pi)\in\theta \wedge j=\pi(i) \wedge j\neq s(j) \\ \qquad s\in\{s\mid ant(s)\in left[\theta,ant(\pi)]\} \\ \tau_{ij}(t+1) = \max[\tau_{\min},(1-\rho)\cdot\tau_{ij}(t+1)], \text{其他}\end{cases}$$

式中，

$$f(\pi) = \frac{[1-(Nstag/50)^{0.2}]}{C(\pi)}$$

$$\tau_{\min} = \frac{1}{10C(\pi)}$$

以上四式中，$ant(\pi)$是构造出解 π 的人工蚁，$\pi(i)$是作业处理序列 π 中的第 i 个作业。$left(\theta,a)$是蚁群种子集 θ 中优先级高于人工蚁 a 的人工蚁集合，$\rho\in[0,1]$为信息素的挥发系数。$C(\pi)$是解 π 的总完成时间，$Nstag$ 是算法运行过程中未能找到更好解的算法迭代次数。此外，函数$f(\pi)$和$\tau_{\min}$的引入，实现了算法的停滞状态脱离机制和信息素踪迹限制机制。

停滞状态脱离机制：

所谓停滞状态是指算法在经过很多次迭代后，当前解仍不能得到改进。在本算法中，函数$f(\pi)$的分子$[1-(Nstag/50)^{0.2}]$随着 $Nstag$ 的增加而从 1 开始平滑下降。这样，θ 中的人工蚁将在其经过的那些很多次迭代都未能改进的解路径上释放较少的信息

素，甚至擦去一部分信息素，将促使人工蚁群脱离停滞状态。

信息素踪迹限制机制：

与最大最小蚁群算法不同，在上面的算法中只设置信息素踪迹的下限 $\tau_{\min}$，以避免算法的过早停滞。由于停滞状态脱离机制的作用，而且信息素踪迹浓度 τ_{ij} 满足 $\lim_{t\to\infty}\tau_{ij}(t) \leqslant 1/C(\pi^{opt})$（$\pi^{opt}$ 是问题的最优解），这时，信息素浓度 $\tau_{\min}$ 的上限是不需要的。在算法中，搜索路径上信息素浓度的初始值为 $\tau_0 = 0.5/C(\pi^{ini})$，其中 π^{ini} 为初始解，它可以随机选择或者由一些简单的解构造算法得到。

5.1.1.3 *局部搜索*（Local Search）

在蚁群优化算法中，问题的解是由蚁群概率性地构造得到的。然而，无论是解 Flowshop 问题的构造式优化算法，还是由随机产生初始解出发的迭代局部搜索算法，其效果相对来说都不是很好。因而，在实际运用中，蚁群优化算法总是与模拟退火、禁忌搜索等邻域搜索算法相结合。一方面，邻域（局部）搜索算法能够进一步改进由蚁群中各个人工蚁构造产生的问题解；另一方面，因为蚁群在构造解时，受反映探索经历的信息素的指引，人工蚁能够为邻域搜索算法构造出好的初始解。

插入式邻域结构：

插入式邻域结构建立在解 π 的插入移动集基础上，解 π 的插入移动可定义如下：

令 $v=(a,b)$ 为解 π 的一对位置序号，插入移动后的新解 π_v 通过从 π 中抽出位于位置 a 的作业 $\pi(a)$ 并将其插入位置 b 来获得。如果 $a<b$，则有：

$$\pi_v = (\pi(1),\cdots,\pi(a-1),\pi(a+1),\cdots,\pi(b),\pi(a),\pi(b+1),\pi(n))$$

如果 a > b，则有：

$$\pi_v = (\pi(1),\cdots,\pi(b-1),\pi(a),\pi(b),\cdots,\pi(a-1),\pi(a+1),\cdots,\pi(n))$$

插入移动集合 V 则由所有这样的位置对 $v=(a,b)$ 组成。解 π 基于移动集 V 的邻域记为：$N(V,\pi)=\{\pi_v: v\in V\}$。邻域 $N(V,\pi)$ 中共有 $(n-1)^2$ 个解（n 为作业数），它满足连通性质，即对于任何初始解 $\pi^{(1)}$，在有限次迭代搜索解序列 $\pi^{(2)},\pi^{(3)},\cdots,\pi^{(r)},\cdots$ 中，一定能搜索到最优解 $\pi^{(r)}$。其中 $\pi^{(i+1)}\in N(V,\pi^{(i)})$，$i=1,\cdots,r-1$。

基于关键路径的邻域结构：

基于关键路径的邻域 $N(Z,\pi)$ 是插入式邻域 $N(V,\pi)$ 的子集，其中，移动集 $Z\subset V$ 建立在解 π 的关键路径和块的基础上。关键路径是流水调度问题网格图中从节点(1,1)至节点(m,n)的一条最长路径，其中节点(i,j)代表在生产单元 i 上处理作业 $\pi(j)$ 的操作工序。网格图是由节点(i,j)及从节点(i,j)指向节点$(i+1,j)$或节点$(i,j+1)$的弧构成的一个有向图，其关键路径可表示为序列 $u=(u_0,u_1,\cdots,u_k)$，块 l 则表示为 $block_l=(u_{l-1},u_{l-1}+1,\cdots,u_l)$。节点 $node(m_l,u_l)$ 代表块 l 中的最后一个操作工序，在块 l 中的所有工序均在生产单元 m_l 处理，而且每个块至少包含两个操作工序。在代表关键路径的序列中，k 表示路径中块的个数。基于上述关键路径和块属性，可以得出以下结论，从而缩小解 π 的可行邻域。

令移动集合 $W(\pi)$ 为：

$$W_l(\pi)=\begin{cases}\{(a,b)\in V:a,b\in\{u_0,\cdots,u_l-1\}\}, & \text{若 } l=1,m_1=1\\ \{(a,b)\in V:a,b\in\{u_{k-1}+1,\cdots,u_k\}\}, & \text{若 } l=k,\\ & m_k=m,W(\pi)=\bigcup_{l=1}^{k}W_l(\pi)\\ \{(a,b)\in V:a,b\in\{u_{l-1}+1,\cdots,u_1-1\}\}, & \text{否则}\end{cases}$$

可得出如下结论：对任何解 $\tilde{\pi} \in N(W(\pi),\pi)$，必有 $C(\tilde{\pi}) \geqslant C(\pi)$。这表明：当前解 π 只有采取属于移动集合 $V \backslash W(\pi)$ 的移动变换操作才可能直接获得改进。

对于高维的排列流水调度问题，邻域结构 $N(V \backslash W(\pi),\pi)$ 仍然太大，它可以进行进一步的压缩。如 Nowicki 和 Smutnicki 定义了如下解移动集合 $Z(\pi,\varepsilon)$ 以及基于该移动集合的邻域 $N(Z(\pi,\varepsilon),\pi)$：

$$Z(\pi,\varepsilon) = \bigcup_{j=1}^{n-1} ZR_j(\pi,\varepsilon) \cup \bigcup_{j=2}^{n} ZL_j(\pi,\varepsilon),\Delta_l$$
$$= (u_l - u_{l-1})\varepsilon, l = 1,\cdots,k,\Delta_0 = \Delta_{k+1} = 0$$

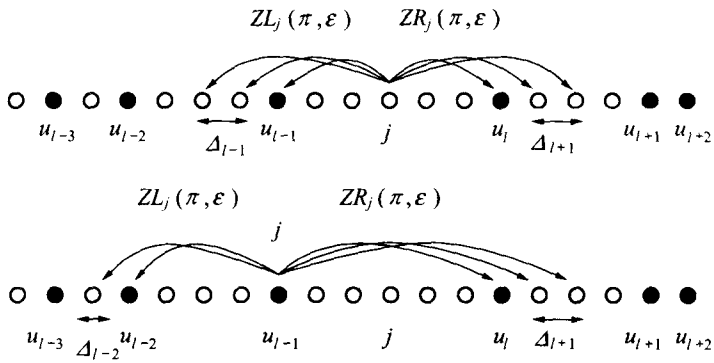


图 5-2 基于关键路径的移动集合示例

上图为移动集合 $Z(\pi,\varepsilon)$ 对解 π 中作业 $\pi(j)$ 的移动操作示例。其中 $ZR_j(\pi,\varepsilon)$，$ZL_j(\pi,\varepsilon)$ 为对作业 $\pi(j)$ 的右移动子集和左移动子集。在此，令移动范围系数 $\varepsilon \in [0,1] = 0.5$。

邻域结构 $N(Z(\pi,\varepsilon)\pi)$ 有如下性质：

$N(Z(\pi,\varepsilon)\pi) \subseteq N(V \backslash W(\pi),\pi) \subseteq N(V,\pi)$，$N(Z(\pi,\varepsilon)\pi)$ 满足连通属性。

在本算法中，邻域搜索过程从蚁群种子集 θ 中各个人工蚁所

构造的解开始，在邻域搜索过程结束之后，搜索过程所找到的质量更好的解将被插入 θ 中，插入的规则同解构造过程。然后，这些解上的信息素浓度也将相应增强。

算法的邻域搜索过程可如下描述：

步骤1 从种子集 θ 中选取一邻域搜索初始解 π。

步骤2 从当前解 π 的邻域中随机选择一个邻域解 $\tilde{\pi}$。

步骤3 如果当前解 π 的邻域解 $\tilde{\pi}$ 满足给定性能指标，则用解 $\tilde{\pi}$ 替代当前解 π 并返回步骤2。

步骤4 如果当前解 π 的所有邻域解都不能改进解 π，或者从上一次获得改进的邻域解 $\tilde{\pi}$ 算起，所搜索的当前解的邻域解个数超过某一设定值，则结束对所选初始解的邻域搜索。否则，返回步骤2继续邻域搜索。

在步骤3中，在当前解的替代规则中定义了一个邻域接受算子。下降算子是常用的邻域接受算子，即当 $C(\tilde{\pi}) < C(\pi)$ 时就用邻域解 $\tilde{\pi}$ 替代当前解 π，这种算子使算法极易陷入停滞状态。随机算子是另一种邻域接受算子，它采用 Metropolis 规则选择邻域解来替代当前解。

Metropolis 规则定义如下：

如果 $C(\tilde{\pi}) < C(\pi)$，则用邻域解 $\tilde{\pi}$ 替代当前解 π，否则以概率 $P_T(\tilde{\pi}) = \exp(-\Delta C/T)$ 替代当前解，其中 $\Delta C = C(\tilde{\pi}) - C(\pi)$。随机算子能够较好地防止搜索算法因过早陷入局部最优而停滞。

5.2 约束优化问题求解

5.2.1 引言

在约束优化问题的求解中，需要找到一组满足约束的最优变量值。这类一般化问题有许多实际应用，如规划调度、资源配置、模式识别和机器视觉等。上面所述的 Flowshop 调度问题实际上

就是一类典型的约束优化问题。

为了解决约束优化问题，可以通过系统完备的方式来进行搜索空间的探索，直到发现一个解或证明该问题无解。为了减小搜索空间，可以将这种完备搜索方法与过滤技巧结合起来。过滤技巧可以缩窄与某些局部一致性相关的变量域。完备性是一个非常好的特性，但是对于某些困难的组合优化问题，它会变得难以处理。尤其是当过滤技巧不能有效减少论域，以使完备搜索可行时。于是，人们提出了非完备的搜索方法，它们以基于概率的方式尽快地找到近似最优解。这些方法随机地探索搜索空间，使用启发式规则把搜索过程引向最有希望的区域。

下面描述了一种用于求解约束优化问题的基于蚁群优化算法的非完备方法。其思想是，通过放置信息素痕迹来寻找搜索空间中有希望的区域。这种信息素作为一种为变量选择分配值的启发式规则，被用来引导搜索。

A. 蚁群优化启发式算法

蚁群优化算法是一种随机方法，它被用来解决各种困难组合优化问题，像旅行商问题、图着色问题、二次配置问题以及车辆路径问题。蚁群优化算法的主要思想是，将问题描述为寻找图中费用最小的一条路径。每只蚂蚁的行为相当简单，以至于它自身通常只能找到品质相当差的路径。发现更好的路径是蚂蚁之间全局合作的突现结果。

人工蚂蚁的行为受实际蚂蚁的启发，它们在图边上留下信息素，并以由信息素痕迹决定的概率选择它们的路径，这些信息素痕迹会逐渐挥发减少。此外，人工蚂蚁有一些不同于实际蚂蚁的特征。它们生活在一个离散的世界（图）中，它们的移动是从结点到结点的转移。而且，它们通常与记忆先前行为的数据结构相联系。在多数情况下，只在已构造完一条完整路径后更新信息素痕迹，而不是在行走过程中，且信息素的累积量通常是路径品质的函数。

人工蚂蚁选择边的概率不仅取决于信息素，还取决于一些基于具体问题的局部启发式规则。

B. 意图及内容概述

当使用蚁群优化算法中的启发式算法解决新的组合优化问题时，一个主要任务是将问题描述为在一个加权图上寻找一条可行的代价最小的路径，其中可行是指满足约束条件。在多数情况下，是以一种先验的方式满足这些约束条件的，即蚂蚁只构造不违反约束条件的可行路径：在行走的每一步，只在可行候选解集（邻域）中选择下一个要访问的点。假定可行候选解集从不为空，以便蚂蚁总能构造可行的路径。例如，在旅行商问题中，约束为每一个城市在任何可行解中只能出现一次。因此，每一个蚂蚁保留一个已访问城市的列表，并在每一次行走时只从未访问的城市集中选择下一个城市。

然而，这种先验约束满足只能用于宽约束问题，此时困难不是找到可行解，而是找到使给定目标函数最优的可行解。本节研究蚁群优化算法解决约束优化问题的能力，即目标不再是最优化有一些约束的目标函数，而是找到满足所有约束的一个值。其主要意图是提供一种处理整类问题的基本工具。

5.2.2 背景

A. 约束优化问题

约束优化问题可由一个三元组(X,D,C)定义，其中X是一个有限变量集；D是从变量$X_i \in X$到值域$D(X_i)$的映射函数，$D(X_i)$为赋予X_i的有限值集；C为约束集，它给出了一些变量之间的关系，限制了这些变量的取值。

集合$A=\{\langle X_1,v_1\rangle,\cdots,\langle X_k,v_k\rangle\}$是一组变量值对，意思是同时将值$v_1,\cdots,v_k$分别赋予变量$X_1,\cdots,X_k$。如果约束$c_i \in C$包含的每一个变量都按$A$赋值后不满足由$c_i$定义的关系，则称解$A$违反约束$c_i$。用$\mathrm{cost}(A)$表示解$A$的代价，它是$A$所违反的约束数。

$CSP(X,D,C)$的解是对 X 中所有变量的完整赋值，且此赋值满足 C 中的所有约束，即其解为代价为零的完整赋值集。

大多数实际约束优化问题中的约束过多，以至于没有解存在。因而，约束优化问题的一般化框架为 Max-CSP。在这种情况下，目标不再是找到一个一致相容的解，而是寻找一个最大满足约束的完整赋值。因而，Max-CSP 的解为一个有最小代价的完整赋值。人工蚂蚁易解决优化问题，在约束优化背景下，它们的目标是最小化违反约束的个数。这种方法可直接扩展到有价值的约束优化问题。

B. 随机二进制约束优化问题

二进制约束优化问题只有二进制约束，即每个约束只包含两个变量。二进制约束优化问题可以随机产生。

一类随机产生的约束优化问题的特征可由四部分$\langle n,m,p_1,p_2\rangle$说明，其中 n 为变量个数，m 为在每个变量域中值的个数，$p_1\in[0,1]$是对连通性的度量(p_1 确定了约束的个数)，$p_2\in[0,1]$是对约束紧度的度量(p_2 决定了对每一个约束不相容值对的个数)。

C. 状态转移

考虑一类约束优化问题时，由于阶参数的改变可以观察到可解性的快速过渡。当从容易求解的约束过少实例向极易证明不一致性的约束过多实例变化时，这些状态转移就会发生。困难的实例通常处于这两种简单实例之间。

为了预测状态转移区域，有参考文献介绍了一类问题的约束性概念 k，并说明当 k 接近于 1 时为严格约束，属于状态转移区域。对于一类随机二进制约束优化问题$\langle n,m,p_1,p_2\rangle$，可定义约束度为 $k=(n-1/2)p_1\log_m[1/(1-p_2)]$。

有人可能会认为状态转移只与完备方法有关，因为它们通常与从可解到不可解实例的过渡相联系，而不完备方法不能发现不

可解性。然而,不同的研究证明,对于诸如搜索的不完备方法中也存在类似的状态转移现象。

5.2.3 Ant-Solver 算法描述

Ant-Solver 用于解决约束优化问题,对于静态组合优化问题,它沿用经典蚁群优化算法的模式,如图 5-3 所示。在每一次循环中,每个蚂蚁构造一个完整的解,然后信息素被更新。当蚂蚁找到一个解,或达到最大循环次数时算法停止迭代。在这一部分,首先定义了人工蚂蚁放置信息素痕迹的结构图,并描述了信息素的初始化;然后,描述了解的构造,以及信息素更新步骤;最后,讨论了参数的设置:

```
Procedure Ant-Solver
    设置参数,并初始化信息素痕迹
    repeat
        for k in 1,…,nb Ants do:构造一个解 A_k
        使用{A_1,…,A_nb Ants}更新信息素痕迹
    until 对于某个 i∈{1,…,nb Ants},cost(A_i)=0
        or 达到最大的循环次数
```

图 5-3 Ant-Solver 算法框架

A. 结构图及信息素痕迹初始化

人工蚂蚁在图上放置信息素,该图称为结构图,其中每一个顶点对应于约束优化问题的一个变量—值对$\langle X_i, v\rangle$。在对应于两个不同变量的任一对顶点之间都有一条边。更正式些,与 $CSP(X, D, C)$相关的结构图是一个无向图 $G=(V,E)$,其中:

$$V = \{\langle X_i, v\rangle \mid X_i \in X \quad and \quad v \in D(X_i)\}$$

$$E = \{(\langle X_i, v\rangle, \langle X_j, \omega\rangle) \in V^2 \mid X_i \neq X_j\}$$

蚂蚁通过在图边上放置信息素互相交流。边$(\langle X_i, v\rangle, \langle X_j, \omega\rangle)$上的信息素量表示为$\tau(\langle X_i, v\rangle, \langle X_j, \omega\rangle)$。直观上,这个信息

素量代表将值 v 赋予变量 X_i，值 ω 赋予变量 X_j 的学习期望度。注意，由于图是无向的，所以 $\tau(\langle X_i, v\rangle, \langle X_j, \omega\rangle) = \tau(\langle X_j, \omega\rangle, \langle X_i, v\rangle)$。

和最大最小蚁群算法一样，我们设置信息素痕迹的下界和上界：$\tau_{\min}$ 和 $\tau_{\max}$（$0 < \tau_{\min} < \tau_{\max}$），目的是通过防止信息素痕迹之间的相对差异变得太极端来支持对搜索空间更大范围的探索。同样，在算法开始时，将信息素痕迹设置为 $\tau_{\max}$，从而在起初的循环中能够搜索空间更大的区域。为了更合适的初始化信息素痕迹并加快收敛，可以引入一个预处理步骤。

B. 解的构造

流程图 5－4 描述了蚂蚁构造完整解的过程。蚂蚁从一个空的值集开始，通过迭代选择要赋值的变量，以及图中对应于该变量的值的点，如此反复，直到所有的变量都被赋值。变量的选择可以某给定步骤进行，而变量值的选择是根据信息素痕迹随机进行的。

```
Procedure 为 CSP(X,D,C)构造解 A
    A←∅
    while |A| < |X| do
        选择一未赋值的变量 X_j ∈ X
        以概率 P_A(⟨X_j,v⟩)选择一个值 v ∈ D(X_j)
        A←A∪{⟨X_j,v⟩}
    end
```

图 5－4 解构造算法框架

注意，应先选变量后选值，使得在每一步候选顶点集及其邻域被限制在单个变量的所有可能值集上，而不是所有未赋值变量的所有可能值集。进行这样的邻域限制是因为，该转移概率的计算量比其他应用问题的计算量（像旅行商问题或二次配置问题）明显大很多。

(1) 变量的选择:在构造解时,变量赋值的顺序非常重要。因为只有在一约束包含的所有变量都被赋值后才能检验该约束是否满足。变量顺序的启发式规则已被广泛研究。一些常用的变量顺序如下:①未来限制越多的变量越先被赋值,即选择(通过约束)与最大数目的未赋值变量相连接的未赋值变量;②已存在约束越多的变量越先被赋值,即选择(通过约束)与最大数目已赋值变量相连接的未赋值变量;③值域最小的先被赋值,即根据已建立的部分解选择有最少数目相容值的未赋值变量;④随机排序,即随机选择一个未赋值变量。

(2) 值的选择:一旦选择了一个变量,蚂蚁必须为其选择一个值。目的是为该变量选择最有希望的值,即在最终解中引起最小冲突数的值。然而,如果直接用已赋值的变量评价冲突数,会更难预测以后为还未赋值变量选择的值的影响。因而,用于引导该选择的启发式规则会非常少,且这些规则通常是基于问题,所以它们不能用于一般的约束优化问题。

对于约束优化问题,蚁群优化算法的主要贡献是提供了用来选择值的一般启发式算法:这种选择是基于概率随机进行的,该概率取决于一个信息素因子(可评价值的学习期望度)和一个启发式因子(可局部评价值的品质)。如果只考虑一只蚂蚁已经访问了点集 A,而它选择的下一个要被赋值的变量为 X_k,则这只蚂蚁选择点$\langle X_j,v\rangle$的概率与相对于其他候选点$\langle X_j,\omega\rangle$($\omega\in D_j$)的吸引能力成正比:

$$P_A(\langle X_j,v\rangle)=\frac{[\tau_A(\langle X_j,v\rangle)]^{\alpha}\cdot[\eta_A(\langle X_j,v\rangle)]^{\beta}}{\sum_{\omega\in D(X_j)}[\tau_A(\langle X_j,\omega\rangle)]^{\alpha}\cdot[\eta_A(\langle X_j,\omega\rangle)]^{\beta}}$$

式中,$\tau_A(\langle X_j,v\rangle)$为$\langle X_j,v\rangle$的信息素因子,$\eta_A(\langle X_j,v\rangle)$为它的启发式因子,参数 α 和 β 决定了它们的相对权重。

与许多蚁群优化算法的一个主要区别是,信息素和启发式因

子不仅取决于候选点$\langle X_j,v\rangle$与上一次访问的点集A中点之间的局部关系,而且取决于候选点$\langle X_j,v\rangle$与整个已访问点集之间的全局关系。实际上,当为X_j选择值时,先前存储在A中的所有赋值是同等重要的。因此,信息素因子$\tau_A(\langle X_j,v\rangle)$取决于在点$\langle X_j,v\rangle$和$A$中已访问点之间的所有边上的所有信息素:

$$\tau_A(\langle X_j,v\rangle) = \sum_{\langle X_k,m\rangle \in A} \tau(\langle X_k,m\rangle,\langle X_j,v\rangle)$$

另外,启发式因子$\eta_A(\langle X_j,v\rangle)$也取决于整个已访问点集$A$,它与将$v$赋予$X_j$时新违反的约束个数成反比:

$$\eta_A(\langle X_j,v\rangle) = \frac{1}{1+\text{cost}(|\langle X_j,v\rangle| \cup A) - \text{cost}(A)}$$

C. 更新信息素痕迹

每个蚂蚁构造完一个完整的解之后,会按照蚁群优化算法更新信息素痕迹。为了模仿挥发,所有信息素痕迹均匀减少,允许蚂蚁忘记相对较差的赋值,循环中用最好的蚂蚁存放信息素。在每次循环末尾,每一条边(i,j)上的信息素量进行如下更新:

$$\tau(i,j) \leftarrow (1-\rho)\cdot\tau(i,j) + \sum_{A_k \in BestOfCycle} \Delta\tau(A_k,i,j)$$

$$\text{if } \tau(i,j) < \tau_{\min}, \text{then } \tau(i,j) \leftarrow \tau_{\min}$$

$$\text{if } \tau(i,j) > \tau_{\max}, \text{then } \tau(i,j) \leftarrow \tau_{\max}$$

式中,$0 \leqslant \rho \leqslant 1$,为挥发参数,*BestOfCycle*为循环中构造的最好解集,即:

$$BestOfCycle = \{A_i \in \{A_1,\cdots,A_{nbAnts}\} \mid \text{cost}(A_i) \text{ 为最小}\}$$

$\Delta\tau(A_k,i,j)$为由得到解A_k的蚂蚁在边(i,j)上放置的信息素量。

$$\Delta\tau(A_k,i,j) = \begin{cases} \dfrac{1}{\text{cost}(A_k)}, if\{i,j\} \subseteq A_k \\ 0, \text{其他} \end{cases}$$

与许多蚁群优化算法相比,蚂蚁不仅在它们走过的路径包含

的边上留下信息素，而且在任意已访问的点对之间所有的边上留下信息素。放置的信息素的量与解的代价成反比，即违反越多的约束，则放置越少的信息素。

这样，本节中基于蚁群优化启发式算法描述了一种用于解决约束优化问题的一般算法 Ant-Solver。该蚁群优化算法可以和任意学习过程相结合，从而改善算法的性能。

5.3 用于二次配置问题求解的蚁群算法

5.3.1 问题介绍

n 阶的二次配置问题是指寻找 n 个设备到 n 个位置的最优配置，其中的设备和位置可有更广泛的含义。T. C. Koopmans 和 M. J. Beckman在 1957 年首先描述了该问题，此后它被用作许多不同实际问题的模型，像大学校园中建筑物的布局、医院中科室的安排、电子电路中最短布线问题，以及磁带中相关数据的排序问题等。

数学上，该问题可由三个 $n \times n$ 维的矩阵定义：

(1) $D = [d_{ih}]$ = (位置 i 与位置 h 之间的)距离矩阵；

(2) $F = [f_{jk}]$ = (设备 j 与设备 k 之间的)流程矩阵；

(3) $C = [c_{ij}]$ = (设备 j 到位置 i 的)线性配置费用矩阵。

通常，矩阵 D 和 F 为整数值矩阵，而设备 j 到位置 i 的线性配置费用 c_{ij} 通常被忽略，因为它不会显著影响待解决问题的复杂度。

在这些假设条件下，排列 $\prod: i \rightarrow \pi(i)$ 可表示从设备 $j = \pi(i)$ 到位置 $i(i = 1, \cdots, n)$ 的一个具体配置。

数据(或原料等，这取决于待解决的问题)转移的费用可被描述为两位置之间的距离与配置到该两位置的两设备之间的流程的乘积，即 $d_{ih} \cdot f_{\pi(i)\pi(h)}$。

因而，为解决二次配置问题，必须找到 $(1, 2, \cdots, n)$ 的一个排

列$\prod$,使得总配置费用最小:

$$\min z = \sum_{i,h=1}^{n} d_{ih} \cdot f_{\pi(i)\pi(h)}$$

为显示目标函数的二次特性,可将问题重新描述如下:

待解决问题意味着确定一个 $n \times n$ 维的矩阵 X(若设备 j 被指派到位置 i,则元素 x_{ij} 为 1,否则为 0),以使:

$$z_{QAP} = \min z = \sum_{i,j=1}^{n} \sum_{h,k=1}^{n} d_{ih} f_{jk} x_{ij} x_{hk}$$

并且满足以下约束条件:

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad (j = 1, \cdots, n)$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad (i = 1, \cdots, n)$$

$$x_{ij} \in 0,1 \qquad (i,j = 1, \cdots, n)$$

式中,矩阵 X 属于 n 阶的排列矩阵集$\prod$。

由于二次配置问题为 TSP 问题的一般化,所以它也是一个 NP 完全问题。

目前可用来寻找最优解的技巧局限于分枝定界和切割平面法。而以现在的硬件条件,不能在可接受的时间内解决高于 25 阶的二次配置问题。

由于这个原因,近年来提出了许多启发式算法,这些算法虽然不能保证所找到的解为最优解,但是能在可接受的计算时间内给出较好的结果。这里拟用蚁群算法来求解二次配置问题。

5.3.2 用于二次配置问题求解的蚁群算法

这里给出的蚁群算法是对基本蚁群算法的改进,它们在几个方面有所不同。具体来说,该改进的蚁群算法通过更新全局记忆结构来引导全局搜索,其中全局记忆结构表示由蚁群在搜索过程中得到的"知识"。同时,还提出了两个与具体问题相关的模块:

局部搜索模块和下界模块。局部搜索模块可使全局搜索集中在局部最优区域,以提高整个过程的效率,而下界模块是蚁群搜索的一个结构组件。

在用于二次配置问题求解的蚁群算法中,每一个人工蚂蚁是具有下述特性的智能体:

(1) 当它选择把设备 j 指派到位置 i 时,它会在连接(i,j)上留下一种称为痕迹的物质(等价于信息素)τ_{ij};

(2) 它以一定的概率为一指定设备选择位置,该概率为"连接(i,j)"的"潜在品质"η_{ij}和在此连接上的痕迹量的函数;

(3) 在构造一个完整的排列时,已经被连接的位置和设备被加以禁止,直到所有的设备都已被指派为止。

该启发式算法使用由 m 个智能体组成的群体来一步步地进行解的构造,为每一个设备指派一个位置。

为了满足蚁群能为每个设备指派不同位置的要求,可先为每个蚂蚁建立一个数据结构,称为禁忌表。禁忌表记忆了已经使用过的设备,并在一次循环结束(即形成一个排列)前禁止蚂蚁为它们分配一个新的位置。一旦排列完成,禁忌表就被清空,蚂蚁又可以自由地选择自己的连接。定义 tabu_k 为第 k 个蚂蚁的禁忌表矢量,$\text{tabu}_k(s)$为该矢量的第 s 个元素(表示由第 k 个蚂蚁选择的占据第 s 个位置的设备)。

现在先来看一下如何计算一种配置下的"潜在品质"η_{ij},以及如何初始化配置(此时还没有痕迹)。此后,该初始化配置将由种群得到的用信息素表示的经验来加以修正。

该基本思想是将一个由有效下界所给出的信息用于完整的问题求解,该下界表示一特定数据对的期望效率。配置品质 η_{ij} 实际上可由该配置的下界的倒数来加以估计。

对于具体的二次配置问题,已经提出了几个下界。最著名的一个是 *GL* 界(由 Gilmore 和 Lawler 分别独立提出),这也是经测试

具有最好的效率/计算费用比率的一个。该下界可通过计算 z_{GL} 值得到：

$$z_{GL} = \min z = \sum_{i,j=1}^{n} (\min \sum_{h,k=1}^{n} d_{ih} f_{jk} x_{hk}) x_{ij}$$

在计算最小值时，还要满足前面所列的三个约束。这相当于，为了定义括弧中的费用项，需要解决 n^2 个线性配置问题，而为了得到 GL 界，还需解决一个线性配置问题。显然，这里有 $z_{GL} \leqslant z_{QAP}$。使用同样的定界策略，也可以得到完整局部配置值的下界。实际上，可假定序列集 $\Phi = \{1,2,\cdots,n\}$ 被划分为两个子集 Φ_1 和 Φ_2，分别对应于已被配置的设备序列和仍未被配置的设备序列。

类似地，假定序列集 $\Lambda = \{1,2,\cdots,n\}$ 也可被划分为两个子集 Λ_1 和 Λ_2，分别对应于已被配置的位置序列和仍未被配置的位置序列。那么可以得到：

$$z_{QAP} = \min z = \sum_{i,h\in\Phi_1} \sum_{j,k\in\Lambda_1} d_{ih} f_{jk} x_{hk} x_{ij} + \sum_{i,h\in\Phi_1} \sum_{j,k\in\Lambda_2} d_{ih} f_{jk} x_{hk} x_{ij}$$
$$+ \sum_{i,h\in\Phi_2} \sum_{j,k\in\Lambda_1} d_{ih} f_{jk} x_{hk} x_{ij} + \sum_{i,h\in\Phi_2} \sum_{j,k\in\Lambda_2} d_{ih} f_{jk} x_{hk} x_{ij}$$

其中，目标函数中的第一项现在是一个已知常数 z_1，而第四项是一个简化的二次配置问题，可使用上式得到一个边界 z_4。通过解决定义在费用矩阵 $[\chi_{lm}]$，$l \in \Phi_2, m \in \Lambda_2$ 上的配置问题，可得到第二项和第三项值的下界 z_{23}，其中：

$$\chi_{lm} = \sum_{i\in\Phi_1} \sum_{j\in\Lambda_1} (d_{il} f_{jm} + d_{li} f_{mj})$$

因而，局部配置完整费用的下界可按下式计算：

$$z_{LB} = z_1 + z_{23} + z_4$$

在这些结果的基础上，为了计算连接 (i,j)，$i \in \Phi_2, j \in \Lambda_2$ 的吸引度，可以简单地为一局部配置进行上式的计算。在该局部配置中，除了已经指定的连接对，还可暂时把设备 i 放到位置 j。因而，可暂时设置 $\Phi_1 = \Phi_1 \cup \{i\}$，$\Lambda_1 = \Lambda_1 \cup \{j\}$，$\Phi_2 = \Phi_2 \setminus \Phi_1$，

$\Lambda_2 = \Lambda_2 \backslash \Lambda_1$，据此计算 z_{LB}，并设置 $\eta_{ij} = z_{LB}$。

在蚁群算法中，基于 η_{ij} 的值和表示痕迹浓度的 τ_{ij} 的值，可使用蒙特卡罗方法基于概率意义构造排列。实际上，定义 $\tau_{ij}(t+1)$ 为与位置 i 和设备 j 相关的痕迹浓度（相当于实际蚂蚁的信息素）。

假设种群中有 m 个蚂蚁，第 $k(k=1,\cdots,m)$ 个蚂蚁将设备 j 配置到位置 i 的概率为：

$$P_{ij}^{k}(t) = \begin{cases} \dfrac{\alpha\tau_{ij}(t) + (1-\alpha)\eta_{ij}}{\sum\limits_{r \notin \text{tabu}_k} (\alpha\tau_{ir}(t) + (1-\alpha)\eta_{ir})}, j \notin \text{tabu}_k \\ 0, \text{否则} \end{cases}$$

其中，$0 \leqslant \alpha \leqslant 1$。

在构造排列时，首先可从位置1开始，通过从所有设备中进行概率性的选择，为其配置一个设备；在第二步，对于第二个位置，从未被指派的设备中概率性地选择一个设备，依次类推。因有 n 个位置，这一过程就可被重复 n 次，而解的构造将被重复 m 次，与群体中蚂蚁的数量相同。

参数 α 允许用户定义痕迹 τ_{ij} 和期望度 η_{ij} 之间的相对重要性。从而，概率 $P_{ij}^{k}(t)$ 是在一连接的期望度（由包含该配置的解的费用下界表示）和痕迹浓度之间的折衷值。

在所有蚂蚁构造完各自的解之后，可按下式进行信息素的更新：

$$\tau_{ij}(t+1) = \rho\tau_{ij}(t) + \Delta\tau_{ij}$$

式中，ρ 是表示痕迹保持特性的系数（$1-\rho$ 代表挥发度），且：

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$$

式中，$\Delta\tau_{ij}^{k}$ 是由第 k 只蚂蚁在构造完自身排列后在连接 (i,j) 上遗留的痕迹量。痕迹的初始浓度 $\tau_{ij}(0)$ 可被设置为一个任意的

小的正值。

系数 ρ 必须 <1，以避免痕迹的无限制累积。对于蚂蚁遗留的痕迹量，计算 $\Delta\tau_{ij}^{k}$ 的不同方法代表了不同的算法实现。在这里，如果第 k 只蚂蚁选择连接(i,j)，则 $\Delta\tau_{ij}^{k}$ 值为 Q/L_k，否则为 0；Q 为当前上界，即在当前代找到的最好解，而 L_k 是第 k 只蚂蚁得到的目标函数值。

这样，最优解（对应于低的 L_k 值）的特性是在连接对上有最多的痕迹，同时对应于小的目标函数的值。

5.3.3 基本算法流程

根据以上介绍，可得到基本算法的流程如下所示：

(1) $t:=0$

初始化信息素矩阵

计算整个问题的上界和下界，以及期望度 η_{ij}

把 m 只蚂蚁放在结点 1

(2) for $k:=1$ to m

repeat{对于每一个位置}

按照计算给出的概率，从尚未被配置的设备中选择一个设备

把选择的设备放到第 k 只蚂蚁的禁忌表中

until 禁忌表满

（该循环被重复 n 次）

end for

for $k:=1$ to m

把解移到它的局部最优，并计算 L_k

局部搜索过程{在这一部分的最后描述}

更新迄今找到的最好的排列

end for

(3) for 每一个连接(i,j) 计算 $\Delta\tau_{ij}$

更新痕迹矩阵

(4) if not (end _ test)

清空所有蚂蚁的禁忌表

go to 2

else

给出最好排列,停止

停止条件通常为最大迭代次数或允许的最大 CPU 时间

算法的性能主要取决于参数 ρ,α,m 的值

这里,可以对蚁群算法的复杂度进行估计。要构造一个完整的排列,必须进行 $O(n^3)$ 次操作。一个完整的迭代(m 只蚂蚁)需要 $O(m \cdot n^3)$ 次操作。当所有蚂蚁都构造完它们的解时,必须更新痕迹矩阵,这需要 $O(n^2)$ 次操作。因而进行算法一次迭代的总的复杂度为 $O(m \cdot n^3)$。

与大多数构造性启发式算法一样,引入局部搜索过程可有效改善蚁群算法的性能,因而可以设计一个两阶段的算法。在第一阶段,按照蚂蚁路径一步一步地进行解的构造。当蚂蚁构造完它的基本排列之后,进行第二阶段——局部搜索,将痕迹加到通常的数据结构中。

这里,局部搜索过程可以是一个简单的确定性过程。如:当蚂蚁得到排列之后,对排列进行所有可能的交换,并评价这些交换的费用,选择最大改善目标函数的交换。

局部搜索过程如下所示:

```
change: = true
while(change: = true) do
    探索由蚂蚁 k 构造的解 s(k)的邻域 s'(k)
    if f(s'(k)) <f(s(k))
    then s(k): = s'(k)
```

```
        else change: = false
    end while
```

对一个解的邻域进行完整探索需要 $O(n^2)$ 次。实际上，邻域中包含 $n(n-1)/2$ 个排列，这可以通过交换成对的元素得到，并且一旦初始化相关数据结构之后，评价费用变量需要一个常数操作时间。这样，对于中大规模问题来说，局部搜索步在计算时间方面也变得相当繁重，还需要设计相应的算法加以改进。

5.4 多选择背包问题求解及应用

背包问题(Knapsack Problem)是运筹学中一个典型的 NP 完全难题。背包问题广泛用于生产实践中，如工厂里的下料问题、人造卫星内的物品装载问题、网络路由中的带宽分配、数据保密中的数据加密等。对该问题求解算法的研究无论在理论上还是在实践上都有一定的意义。对于背包问题已有很多求解方法，如回溯法、遗传算法、分枝一定界法等。这里将总结一下如何使用蚂蚁算法求解多选择背包问题。该方法具有正反馈的寻优特性，再结合变异参数，使算法既有较快的求解速度，又有较高的求解精度。实验结果表明，采用此算法能快速有效地解决背包问题。

5.4.1 多选择背包问题基本模型

背包问题有多种描述形式。一般的整数背包问题可以描述如下：一个人上山带一个背包，他可携带物品的总质量的限度为 Vkg，设有 n 种物品可供选择装入背包中，这 n 种物品编号为 1，2，…，n。已知第 i 种物品每件质量为 W_i，每件价值为 P_i，携带个数为 X_i（$X_i \geqslant 0$ 且为整数）。问：此人应如何选择携带物品（各几件），使总价值 Z 最大？

问题模型可以表述为：

$$\max Z = \sum_{i=1}^{n} P_i X_i$$

$$s.t.\begin{cases}\sum W_i X_i \leqslant V \\ \qquad\qquad (i \in \{1,2,\cdots,n\}) \\ X_i \geqslant 0 \text{ 且为整数}\end{cases}$$

5.4.2 蚁群系统求解多选择背包问题

如前所述,蚁群算法模拟的是蚂蚁群体的行为特性,利用蚁群在搜索食物源时所具有的寻优能力来解决生产生活中的优化难题。

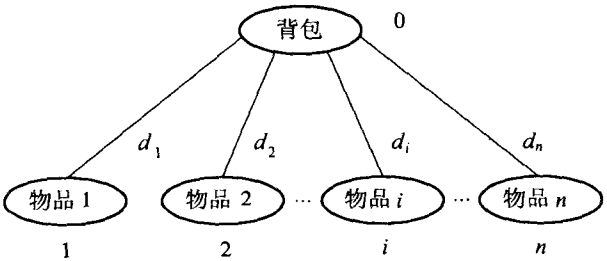


图 5-5 多选择背包问题的描述

对于基于蚁群算法的多选择背包问题求解可以用图 5-5 描述。设节点 0 代表背包,节点 1 ~ n 分别代表各物品,从节点 0 到任意节点 $i(i \neq 0)$ 的路径的权值为 $d_i = W_i/P_i$。可以将背包(节点 0)看作蚂蚁寻优的起点,任一物品 $i(i \neq 0)$ 看作可供蚂蚁选择的一个食物源,d_i 可以理解为从寻优起点到食物源 i 的距离。对于任意一只蚂蚁 k,其从位置 i 转移到位置 j 的概率:

$$P_{ij}^{k} = \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum \tau_{is}^{\alpha}\eta_{is}^{\beta}} \quad (i = 0, j \in \{1,2,\cdots,n\})$$

τ_{ij} 表示路径中留下的信息素强度,该参数表明了系统后天演化过程中从位置 i 转移到 j 能使系统得到的受益度;η_{ij} 属于先天性的启发式,表示该转移对于蚂蚁 k 的吸引度。在求解背包问题

时 $\eta_{ij}=1/d_j$。α 为路径中信息素强度的重要性($\alpha\geqslant 0$);β 为吸引度的重要性($\beta\geqslant 0$)。由于背包问题属于特例,每只蚂蚁从寻优起点(节点 0)出发只需一步就可到达 $1\sim n$ 当中的任意一个食物源,因此在上式中有($i=0,j=1,2,\cdots,n$)。

整个背包问题的求解过程是由多只蚂蚁完成的。每当一只蚂蚁以较大的概率选中食物源 j 时,如果 $V-\sum_{s=1}^{n}W_sX_s\geqslant 0$,变量 X_j 将加 1;否则将给从 i 到 j 的路径以惩罚值,以使后续的蚂蚁在本次背包问题求解中不再选择该路径。当所有目的节点都受罚以后将结束一次求解过程。这时可记录最优解,然后对路径中信息素强度进行更新,其更新的方程式为:

$$\Delta\tau_{ij}=QW_jX_j$$

$$\tau_{ij}=(1-\rho)\tau_{ij}+\Delta\tau_{ij}(i=0,j\in\{1,2,\cdots,n\})$$

式中,ρ 为挥发系数,Q 为正常数,X_j 为经过该路径的蚂蚁数量。

真实的蚁群在寻优时,蚂蚁会以较大的概率朝着概率值较大的方向移动,但也有蚂蚁会选择别的方向。正是这种特性使得真实的蚁群既能快速找到最佳路径,又不会限于局部最优。这里,在背包问题的求解中使用了变异参数来模仿真实蚁群的这种特性。令 *mutation* 为变异系数(0.5~1),设蚂蚁 K 将要从 i 转移到的目标节点为 j,则 j 由下面的式子决定:

$$j=\begin{cases}p,\text{若 random}\leqslant mutation\\ m,\text{否则}(m\text{ 为随即选择的其他目标地点})\end{cases}$$

目标节点 p 满足 $\tau_{jp}^{\alpha}\eta_{jp}^{\beta}=\max\{\tau_{is}^{\alpha}\eta_{is}^{\beta}\}$,$s\in 1,2,\cdots,n$,*random* 为 $[0,1]$ 上的随机数。

5.4.3 求解背包问题的具体蚁群算法框架

步骤 1

(1) $NC\leftarrow 0$(迭代步数或搜索次数);

(2) 各 τ_{ij} 和 $\Delta\tau_{ij}$ 初始化;

(3) $k \leftarrow 1$。

步骤2

(1) 对蚂蚁 k 计算从源点(节点0)到各目标节点($1,2,\cdots,n$)的最大概率值。

(2) 若蚂蚁 k 未发生变异,则沿着较大的概率方向移动;否则随机选择其他方向移动。

(3) 对于蚂蚁 k 选择的节点,若未使背包超重,则该节点的记数加1;否则设置惩罚值,以使后续的蚂蚁不再选择该路径。

(4) $k \leftarrow k+1$。

步骤3

若还有目标节点没有受罚且 $k < ANT\_NUMBER$,则返回步骤2;否则记录当前最优解,更新路径信息素强度。

步骤4

若 NC 还没有达到最大迭代次数并且解的寻优值还在变化,则 $NC \leftarrow NC+1$,$k \leftarrow 1$,转到步骤2;否则,输出最优解,终止程序。

本算法的复杂度为 $0(NC \cdot ANT\_NUMBER \cdot 2(n+1))$,其中 NC 为迭代次数,ANT_NUMBER 为算法中使用的蚂蚁数量,n 为物品种类。

5.4.4 仿真结果分析

针对多选择背包问题,这里将精确求解方法(回溯法)和蚂蚁算法进行了对比。实验中所用数据为随机生成,各 W_i 在1~100内随机生成,物品种类取值为5~20。背包容量取值为

$$V = 0.8\sum_{i=1}^{n} W_i X_i$$

对于给定的物品数量,用随机生成的数据进行实验,并针对求解时间求平均值。通过对比,发现蚂蚁算法求解速度很快。表5-1是在解值相同的情况下,蚂蚁算法和回溯算法在求解时间

上的对比。

表 5-1 蚂蚁算法和回溯算法在求解时间上的对比

物体个数	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
回溯算法	<1ms					240ms	2s	3s	7s	16s	33s	42s	88s	190s	393s	620s
蚂蚁算法	<1ms												40ms	50ms	56ms	66ms

背包问题属于 NP 难题，对于背包问题在规模较大的情况下，精确求解将会产生组合爆炸，求解过程将会花费漫长的时间。因此，怎样能够快速有效地求解背包问题长期以来一直是人们研究的课题。蚂蚁算法向人们展现了一种随机型的启发式优化方法。在以上仿真中可以发现，蚂蚁算法求解速度很快，通过取适当的变异系数（取较小的变异系数，扩大求解范围），还可以求得比较精确的解。蚂蚁算法正是在求解速度和求解精度之间寻找平衡点的一种算法，是一种具有“柔性”的算法。

第6章　蚁群算法的典型应用

如前面所述，蚁群算法的应用范围已经遍及计算机、机器人研究、电力系统、通信、化工领域等。本章将从蚁群算法的典型工程应用领域出发，全面综述蚁群算法在其中的应用成果，并作适当评述。

6.1　机器人领域

这里，以基于蚁群算法的自由飞行空间机器人路径规划为例，说明蚁群算法在机器人领域的应用。

6.1.1　总体思路

在未来空间资源的开发利用中，空间机器人将发挥举足轻重的作用。自由飞行空间机器人（Free Flying Space Robot，FFSR）是一种新型的智能机器人，它由机器人本体（卫星）和其搭载的机械手组成。由于FFSR的本体内携带气体推进器，它可以在空间微重力环境下自由飞行或浮游，从而扩展了机器人的工作空间，因此，它将代替宇航员从事各种舱外作业，成为今后空间机器人的主要研究方向之一。FFSR的燃料有限，为了提高其工作效率以及延长其在轨寿命，对其飞行运动的研究具有重要意义。路径规划对于FFSR来说是非常重要的，但机器人的路径规划问题属于一类复杂的非线性优化问题，传统的优化方法在该类问题的求解中缺乏足够的鲁棒性。

下面描述一类基于蚁群算法的机器人路径寻优策略。它将蚁群算法进行适当改进，使之适用于FFSR的路径规划，经过蚁群的协同工作，以找到一条优化路径。

6.1.2 路径的生成(Building of Path)

假设 FFSR 所处的位置为 S,目标点为 G。S、G 之间有一些障碍物,如图 6-1 所示。FFSR 需要寻求一条从 S 点到 G 点的既短又安全的一条路经。

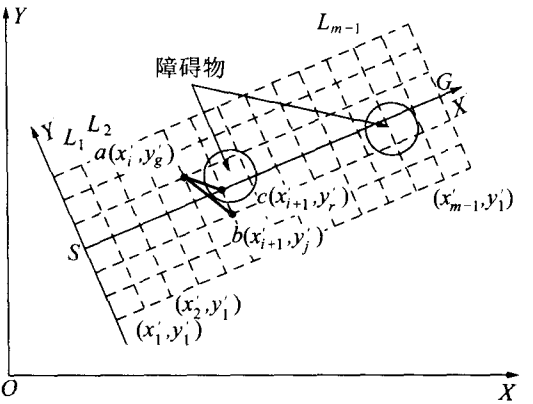


图 6-1 机器人路径产生过程示意图

FFSR 起始点 S 在笛卡儿坐标系 $O-XY$ 下的坐标为(x_s, y_s)。现在先以 S 点为坐标原点,SG 为 X'轴,垂直于 SG 的直线为 Y'轴,建立新的坐标系 $S-X'Y'$。然后将线段 SG 进行 m 等分,在每个等分点作 SG 的垂线,就得到线段 $L_1, L_2, \cdots, L_{m-1}$,再以 X'轴为中心,将每条线段进行 $2n$ 等分,每条垂线上就有$(2n+1)$个点。这样,在避障区域内,就有$(m-1)(2n+1)$个路径点,即:

$$L_1(x_1', y_1')L_1(x_1', y_2')\cdots L_1(x_1', y_{2n+1}')$$

$$\cdots\cdots$$

$$L_{m-1}(x_{m-1}', y_1')L_{m-1}(x_{m-1}', y_2')\cdots L_{m-1}(x_{m-1}', y_{2n+1}')$$

其中,$L_i(x_i', y_j')$表示第 i 条垂线上的第 j 个点,则从起始点 S 到终点 G 的路径可以表示为:

$$Path = \{S, L_1(x'_1, y'_{k1}), L_2(x'_2, y'_{k2}), \cdots, L_{m-1}(x'_{m-1}, y'_{k(m-1)}), G\}$$
$$(ki = 1, 2, \cdots, 2n+1)$$

新坐标系 $S-X'Y'$ 到原坐标系 $O-XY$ 的转换公式为：

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} + \begin{bmatrix} x_s \\ y_s \end{bmatrix}$$

式中，α 表示 OX 与 SX' 的夹角。

垂线 L_i 上的路径点 $a(x'_i, y'_g)$ 到下一个垂线 L_{i+1} 上的路径点 $b(x'_{i+1}, y'_j)$ 的距离可以表示为：

$$d_{ab} = \sqrt{\left(\frac{|SG|}{m}\right)^2 + (y'_j - y'_g)^2}, \quad j, g = 1, 2, \cdots, 2n+1$$

如果线段 ab 与障碍物相交或相切，则令其距离为 ∞，如线段 ac。

这里，第 k 只蚂蚁的路径长度为：

$$L_k = \sqrt{\left(\frac{|SG|}{m}\right)^2 + (y'_{k1} - 0)^2} + \sum_{ki=1}^{m-2} \sqrt{\left(\frac{|SG|}{m}\right)^2 + [y'_{k(i+1)} - y'_{ki}]^2} + \sqrt{\left(\frac{|SG|}{m}\right)^2 + [y'_{k(m-1)} - 0]^2}$$

6.1.3 相关蚁群算法的定义

由于 FFSR 避障路径规划问题与 TSP 问题之间存在差异，因此在此所用的蚁群算法与经典论文中的蚁群算法既有相似之处，也有不同之处。

相似之处在于：在 TSP 问题中，旅行商遍历各城市时要寻求的总路径长为最短；而在 FFSR 路径规划中，FFSR 所走的路径也需为最短。

而不同之处在于：

(1) TSP 问题中旅行商的路线是一条闭合路径，需要走完全部的城市；而 FFSR 则无需遍历全部节点，只需从出发点出发到达目标点即可。

(2) TSP 问题中蚂蚁根据它的总路径长度来更新信息素；FFSR 则根据目标函数来更新信息素，其中，目标函数中不但包含蚂蚁走过的路径长度信息，还应包含避障安全信息。

(3) TSP 问题中蚂蚁必须记住已走过的节点；而 FFSR 路径规划中，FFSR 无需记忆，只需选择下一条垂线上的节点即可。

6.1.3.1 目标函数的建立

假设现共有 q 个障碍物，每个障碍物的大小可表示为圆心位于(X_j', Y_j')，半径为 r_j 的圆，垂线 L_i 上的节点(x_i', y_{ki}')到障碍物的距离可表示为：

$$d = \sqrt{(x_i' - X_j')^2 + (y_{ki}' - Y_j')^2} - r_j$$

由于所定义蚁群算法中的信息素是根据目标函数的值来更新的，这样，目标函数的选择还应该考虑到具体问题的特征，即在寻求路径最短的同时还能够安全避开障碍物。从这两点出发，用蚂蚁所走过的路径长度和路径上选定的离散点到最近障碍物的距离作为参数确定目标函数，则所定义的目标函数计算公式如下：

$$F = L_k + \delta \sum_{i=1}^{m-1} \frac{1}{d_{imin}}$$

式中，L_k 表示第 k 只蚂蚁所走过的路径长度，d_{imin} 表示节点到最近障碍物的距离，δ 为避障系数。δ 越大，FFSR 的安全系数就越高。

选定的节点(x_i', y_{ki}')到最近障碍物距离的计算公式如下：

$$d_{imin} = \min\{(\sqrt{(x_i' - X_1')^2 + (y_{ki}' - Y_1')^2} - r_1), \cdots, (\sqrt{(x_i' - X_q')^2 + (y_{ki}' - Y_q')^2} - r_q)\}$$

6.1.3.2 路径点的选择

假设蚂蚁从垂线段 L_i 上的节点 α 到下一个垂线段 L_{i+1} 上任意节点 b 的时间相等，与距离无关，那么全部蚂蚁将同时到达目标点，同时完成一次循环。

如果在 t 时刻，蚁群移动到垂线段 L_i 处，设 $b_j(j=1,2,\cdots,2n+1)$ 为 t 时刻线段 L_i 上 j 节点处的蚂蚁数，则蚂蚁总数 h 可以表示为 $h=\sum_{j=1}^{2n+1} b_j$。

设 $\tau_{ab}(t)$ 表示 t 时刻在路径线 ab 上残留的信息量。在初始时刻，各条线上的信息量相等，设 $\tau_{ab}(0)=C$（C 为常数），$\Delta\tau_{ab}=0$。蚂蚁 k 在运动过程中，可根据各条路径线上的信息量决定转移方向。$P_{ab}^{k}(t)$ 表示 t 时刻蚂蚁 k 由位置 $a(x_i', y_g')$ 转移到 $b(x_{i+1}', y_j')$ 的概率：

$$P_{ab}^{k}=\begin{cases}\dfrac{\tau_{ab}^{\alpha}(t)\eta_{ab}^{\beta}(t)}{\sum_{0}\tau_{ab}^{\alpha}(t)\eta_{ab}^{\beta}(t)}, & b\in allowed\\ 0, & \text{否则}\end{cases}$$

式中，η_{ab} 表示线段 ab 上的能见度，α 表示信息素的相对重要性（$\alpha\geqslant 0$），β 表示能见度的相对重要性（$\beta\geqslant 0$）。

能见度 η_{ab} 为 a,b 点距离的倒数，即 $\eta_{ab}=\dfrac{1}{|d_{ab}|}$。

6.1.3.3 信息素的更新

随着时间的推移，先前留下的信息素将逐渐消失。这里，可用参数 $\rho(0\leqslant\rho<1)$ 来表示信息素的持久性，则 $1-\rho$ 表示信息素的消失程度。经过 m 个时间单位，蚂蚁从起始点 S 到达目标点 G，则各条路经上的信息量可根据下式进行调整：

$$\tau_{ab}(t+m)=\rho\tau_{ab}(t)+\Delta\tau_{ab}$$

$$\Delta\tau_{ab}=\sum_{k=1}^{h}\Delta\tau_{ab}^{k}$$

式中，$\Delta\tau_{ab}^{k}$表示第 k 只蚂蚁在本次循环中，在线段 ab 上留下的单位长度轨迹上的信息素均值，可通过下式进行计算：

$$\Delta\tau_{ab}^{k}=\begin{cases}\dfrac{Q}{F_k}, & \text{若第 } k \text{ 只蚂蚁在本次循环中经过 } ab\\ 0, & \text{否则}\end{cases}$$

式中，Q 为常数，F_k 表示第 k 只蚂蚁在本次循环中的目标函数值。

6.1.4 所定义的蚁群算法流程

这里所用的蚁群算法主要步骤为：

步骤 1：令时间 t 和循环次数 NC 为零，设置最大循环次数 $NC_{\max}$，令每个线段上的信息量 $\tau_{ab}(t)=C$，且 $\Delta\tau_{ab}=0$，同时将全部蚂蚁置于起始点 S。

步骤 2：启动所有蚁群，对每只蚂蚁 k，按以上所定义的概率用轮盘法选择下一条垂线上的节点并前进。

步骤 3：重复步骤 2，直到蚁群到达目标点 G。

步骤 4：令 $t\leftarrow t+m$；$NC\leftarrow NC+1$；计算各蚂蚁走过的路径长度 L_k 和目标函数值 F_k，记录当前最优解。根据公式 $\tau_{ab}(t+m)=\rho\tau_{ab}(t)+\Delta\tau_{ab}$，$\Delta\tau_{ab}=\sum_{k=1}^{h}\Delta\tau_{ab}^{k}$来更新每个线段上的信息残留量。

步骤 5：如果蚁群全部收敛到一条路径或循环次数 $NC\geqslant NC_{\max}$，则循环结束，输出最佳路径。否则回到步骤 2。

以上算法中，通过调整避障系数，可以得到不同的优化轨迹，从而扩展机器人对具体问题的适应性。同时，该算法也有利于并行执行和应用，并且具有较强的鲁棒性。相关仿真结果显示，蚁群算法在解决路径规划等优化问题方面有良好的前景。

6.2 交通运输规划问题求解

这里，以上海市内河航道规划问题求解为例，介绍蚁群算法

在交通运输规划问题求解中的应用。

6.2.1 总体思路

这里所针对的分析对象为上海市航运网的随机优化问题求解。其运算网络包含 33 个节点,38 条河段,每节点有相应的运输任务,同时每河段有相应的运输能力。这里的运输任务与运输能力都是对未来的一个预期,有一定的随机性。在国内相关研究人员的努力下,对蚁群算法做了相应的改进,并配合随机分布技术,以上海市整个内河航道和集装箱运输为研究对象,对内河航道进行了合理规划,得出了上海市内河集装箱集散系统合理的分配方案,并提出为满足该合理系统所需进行的相应的河道改造重点。

6.2.2 相应蚁群算法的设计思路

这里所采用的分析对象为上海 2005 年的运输任务及航运网。在完成运输任务的同时,要求得出最优航道的选择情况和航道满载的概率,以期求得各河段相应的改造概率,从而保证该运输网络的畅通。实际所采用的网络模型由上海市现有内河航运系统抽象而成,如图 6-2 所示。

上海市内河航运网中,所需要考虑的因素与 TSP 问题有较大的出入:

(1) 内河航运网不是完全的带权图,其规划最终目的也不是找一条代价最小的 Hamilton 回路;

(2) 内河航运规划牵涉到许多随机的因素,这里只能先主要考虑河道运输能力的随机变化和各地集装箱数量的随机变化这两种因素;

(3) 河段有运输能力的限制,而且内河航运网还有明确的运输任务要求。

因此,要利用蚁群算法来求解内河航运网的优化问题,就要求对蚁群算法做一定程度的改进,而不能限于 TSP 模型的框架中。

这里,首先不要求蚂蚁寻找出完整的 Hamilton 回路, 而只需

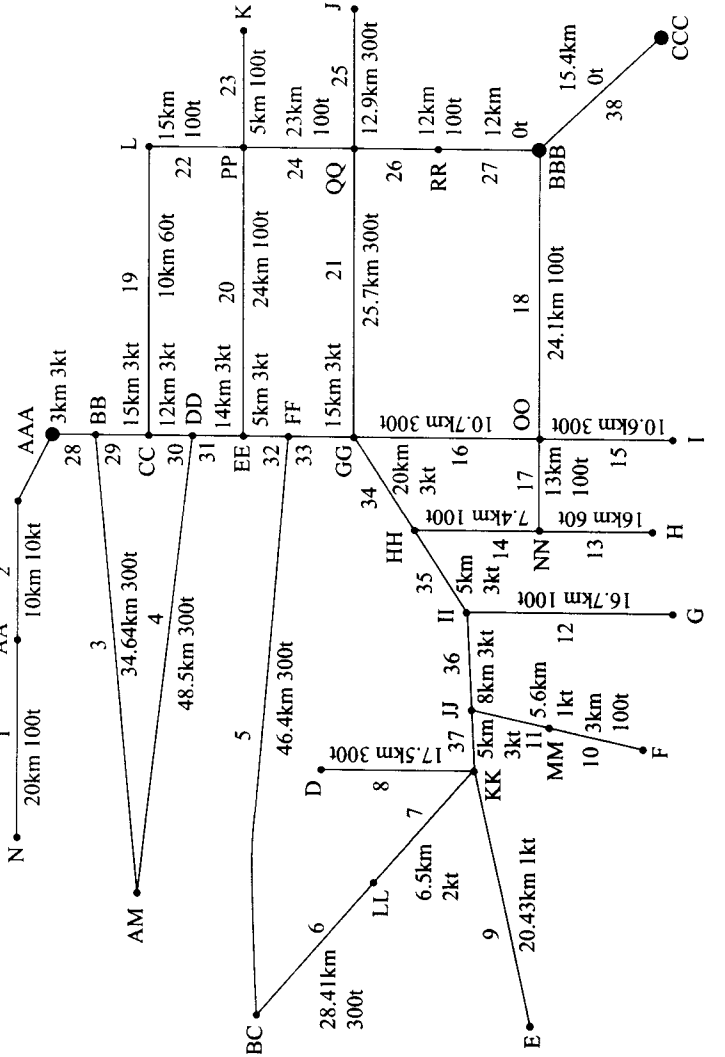


图 6－2 上海市内河航运网络图

作出一定的限制，让其在所有具有运输能力的路径中寻找出不重复两点之间的最短距离。这样，就扩大了蚁群算法的应用范围，使之能适应许多场合的优化问题。当涉及到的路径不多时，在初始化阶段所置的蚂蚁数可以适当增多。这样既可以加强蚁群搜索的范围，也可以使搜寻结果更优。

蚂蚁在旅程中，会记录它所通过的每个点。当记忆中的点有重复时，也即出现了环路，这时就宣告该蚂蚁搜索失败，程序将其搜索结果自动从数据库中删除。显然，在一条通路中如果包含环路，则肯定不是两点间的最佳通路。这样，某只蚂蚁到达目的点的时候，其旅程也告结束。另外，蚂蚁寻找路径，还需考虑各路径的实际通过能力，当某路径的能力耗尽时，则应把该路径视为截断。

6.2.3 具体的蚁群算法流程

根据以上所述的总体原则，可设计相应的算法流程如下：

begin

分配各点及各路径的编号；

for $i = 1$ to 33

初始化阶段；

选择目的地；

在 i 节点置 m 只蚂蚁；

每条路径上信息素的初始化；

该点货物总数的初始化（当前基数加上未来可能变化的随机量）；

每条河段运输能力的初始化（固定能力加上随机变化）；

运算阶段；

$c = 0$

while 没到目的地 and $k < M$（M 是预先设定的一大数，防止死循环）

```
        for c = 0 to n
            蚂蚁从上次的末节点出发,寻找下一路径;
            将蚂蚁出发点的运输任务分配到其通过的河段;
            判断河段是否已经满载,如果是,则修改数据表,视
            此河段为断路;
            局部更新,根据该蚂蚁该次通过路径的长短,更新其
            上的信息素数量,并记录其长度及节点编号;
        next
        end for
    c = c + 1
    next
    打印各点到目的点的路径及运输任务的分配;
    求出各河段的满载率并打印
    end for
    对整个网络进行统计,得出每条河的利用率
    end
```

6.2.4 实例运算结果

计算结果如表6-1所示,表中 i 为计算次数。

由表6-1可见,河段1与2虽然运输任务差不多,但由于能力相差悬殊,所以利用率相差较大。相对而言,河段16~18运输任务较重,承担着许多节点的运输任务,而且运输能力有限。虽然2005年在CCC深水港分配的任务不多,但这些河段利用率已经比较高。随着CCC深水港的建设,这些河段将很快满载而需要改造。河段19比较特殊,其运输能力较小,东面是外高桥,考虑到以后河段的部分运输能力将为外高桥所占用,而且其周边地域广阔,对运输量要求较大,所以自然最先达到饱和状态。目前,上海市对这一河段正在进行必要的改造。由于河段19的满载,剩余的运输量将自然转移到与其相邻的比较合理的河段20。对于河段21,因

为不管从哪点到港口，都不是最佳的路径；而且，周围河段的运输能力都没有完全利用，所以，其运输量暂时为0。等到相邻河段16、18、20等的运输量增加而不能满足时，方才选择河段21。河段26、27在运输网中的位置很关键，鉴于目前的改造情况，运算时预先设定其运输能力为100。和河段16～18等一样，在以后的日子里，河段26、27的运输量也将大幅度上升。河段38是CCC深水港与各点的唯一通道，随着深水港的建成，应该将其建设成一条运输量很大的河段方能满足需要。

表6-1 代表性河段的满载率及满载概率表

河段代号	满载率/%										满载概率/%
	$i=1$	$i=2$	$i=3$	$i=4$	$i=5$	$i=6$	$i=7$	$i=8$	$i=9$	$i=10$	
1	37.08	37.41	37.06	38.65	37.46	35.96	38.17	34.55	35.11	36.22	0
2	1.84	1.55	0.95	1.27	1.53	1.43	1.51	1.52	1.63	0.99	0
16	19.44	19.29	18.15	19.17	19.57	18.47	18.89	18.19	18.87	19.24	0
17	31.21	31.78	31.58	31.09	31.46	31.72	31.64	30.78	30.45	31.12	0
18	39.64	37.99	37.98	38.11	38.47	38.34	38.67	39.17	39.21	38.69	0
19	100	100	100	100	100	100	100	100	100	100	100
20	35.82	34.12	35.16	36.07	38.07	34.98	34.99	37.09	35.82	34.56	0
21	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0
22	98.10	100	97.68	99.01	99.80	100	97.98	98.54	98.67	100	30
26	39.81	38.98	38.99	39.47	39.18	39.74	38.84	39.11	39.17	39.11	0
27	32.03	32.04	32.01	30.38	32.85	31.08	30.82	30.89	31.65	32.35	0
28	12.89	14.04	13.58	13.64	13.89	12.98	14.01	14.18	12.89	14.18	0
38	70.47	70.54	70.88	70.70	70.67	71.00	70.85	70.31	70.11	70.03	0

6.3 集成电路布线设计

这里以基于蚁群系统的两端线网布线方法为例，说明蚁群算法在集成电路布线设计中的应用。超大规模集成电路(VLSI)物

理设计(Physical Design),也称为布图设计(Layout Design)。在布图设计中,需要根据电路和工艺的要求完成芯片上单元或功能块的安置(布局),然后实现它们之间的互连(布线)。两端线网的布线问题,可看成是在一个有网格的平面中,求取两个端点之间绕过障碍的最短路径。在 VLSI 布线设计中,金属布线层上的电路元件、已经布好的互连线、甚至待连接的电路端点都可看作是障碍,如何绕过这些障碍找到互连的路径,或找到对今后布线最有利的路径,并达到总的线长及时延的最小化,是需要解决的问题。对于绕障碍寻找目标路径的布线问题,目前已经有了一些算法解决方案,如迷宫算法、线探索布线、平面扫描法、图论方案、计算几何法、波前端法以及动态搜索法等,但这些方法都有一定的缺点。

这里所介绍的基于蚁群系统的两端线网布线方法,为解决这类布线问题提出了一种新的思路。实验结果表明:在参数选择适当的情况下,均能很快地找到最短路径。

6.3.1 问题描述

网格的大小直接与制造工艺有关,一般可取最小线宽与最小线距之和,每个网格的长度可定义为单位 1。由网格决定的可布线区域的边界必为直线段,称之为直线区域(Rectilinear Region)。在直线区域中,两端点布线问题就转化为寻找最短路径的问题。在单层布线平面中,假定 R 为 $n \times n$ 的网格图,$O = \{O_1, O_2, \cdots, O_k\}$为网格内的不可布线区域集合,$O_k$ 为相关障碍(Obstacle),那么 $G = R - O$ 即为可布线区域。对于给定的两线网端点集合 $V = \{s, t\}$,如果存在 G 的一个子集 C,能够使得源点 s 与目标点 t 在 G 中相连接,即 C 中至少存在一条连接 s 与 t 的路径,则称 C 为 G 的连接图(Connection Graph):当能在 C 中实现 s 与 t 在 G 中的最短路径连接时,则 C 成为 V 在 G 中的强连接图 SC(Strong Connection Graph)。寻找在 G 中的强连接图 SC 以代替 G 可使 SC 的规模 $|SC|$ 大大地小于 G 的规模 $|G|$,从而简化问题,

提高效率。在有网格的布线平面中，点与点之间的距离是用曼哈顿距离 MD（Manhattan Distance）来表示的，若用坐标表示，则 $MD = |x_1 - x_2| + |y_1 - y_2|$。显然，一旦两端点的位置确定，$MD$ 也就确定了。在无障碍的情况下，最短路径 $SP = MD$；在有障碍的情况下，可构造一个图，使得边长以分形距离 FD 为准。由于 FD 更接近实际的最短路径，当存在两点最短路径时，可构造两端绕障碍网格布线的强连接图，从而可以减小问题规模、节省空间、提高布线效率。

6.3.2 问题的连接图模式

直线区域 R 如图 6－3 所示，对于给定的两端点 s,t（用 • 表示），假如在 G 中存在路径，必定可以让 s 与 t 沿着障碍边界（或边界）从 s 找到 t。但这往往并不是最短路径，为了方便地找到最短路径，我们可以构造连接图 $C = (V,E)$：其顶点 $v \in V$ 是线集合的交点，这个线集合是与障碍 O_k 各边紧相邻的直线（代表障碍边）和经过 s,t 点的直线的集合；各边 $e \in E$ 的权值则为两交点间的距离。因为各点间是可能连通的，并存在最短路径，所以必存在强连接图 SC。

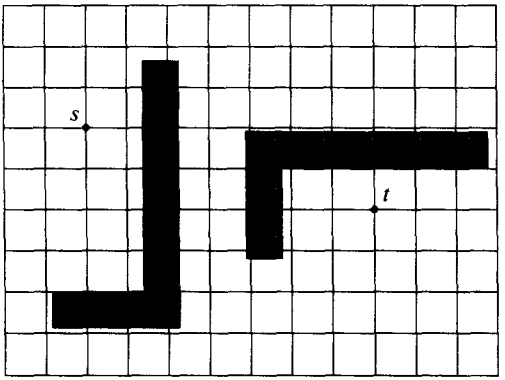


图 6－3 两端绕障碍网格布线的强连接图说明

这里的初始数据为端点数、障碍数以及各障碍的网格位置信息，先排序 s，t 和所有障碍角点的横坐标和纵坐标（定义为紧挨实际障碍的、可布的网格位置坐标），它们所在直线的所有交点就构成了连接图的交点集合。同时，用矩阵记录下该交点集合相应的原始网格位置信息和状态信息（可以分为障碍角点、障碍边点、障碍内部点和生成的可布点等情形，在布线过程中还存在此次寻径过程中蚂蚁已经走过的状态，双层问题中还包含点所在层信息等）。

6.3.3 基于连接图模型的蚁群算法实现

对连接图的交点集合中的所有交点给出适合的初始值，形成初始气味矩阵 T_0（其值是各节点到目标节点 t 的分形距离的 FD^a，$a<0$）。在应用蚁群算法搜寻目标路径点的过程中，气味矩阵的值将随局部气味更新和全局气味更新的规则而变化。这就将分布式地反映可行解信息，使每只蚂蚁在行进过程中运用状态转移规则 S 在可能的邻接点集合 $J(i)$ 中进行选择（还可以适当添加一些启发式规则，如在规则 S 的相同值下优先选择不改变路径前一线段方向的点等）。$J(i)$ 是根据状态矩阵信息求解的，直到到达目标节点 t。当所有蚂蚁都搜索完目标点后，应对寻找到的所有已走过路径中的最短路径上的所有节点进行全局性的信息素更新，并且重复以上过程，直到求出优化解。

因为这里是网格布线，所以连接图可用 $T(V)$ 表示。设蚂蚁位于 $T(V)$ 的节点 i，从 V 中找出可能的下一点集 $J(i)$，这里 $J(i)$ 是其相邻节点集，已经排除不符合状态的节点。蚂蚁按照下式给出的状态转移规则，从可能的下一点集 $J(i)$ 中选择下一点 j 前进。

$$j=\begin{cases}\underset{u\in J(i)}{\operatorname{argmax}}\{[\tau(u)]\cdot[\eta(i,u)]^{\beta}\}, & \text{若 } q\leqslant q_0\\ S_{old}, & \text{其他}\end{cases}$$

式中，$\tau(u)$是节点 u 上存储的蚂蚁分泌的信息素；$\eta(i,u)$是节点 i 和 u 间距离的倒数[这里的目标是求最短路径，在其他问题中，$\eta(i,u)$也可以是其他性能参数(如费用)的反增长函数]；β 用于权衡信息素浓度和局部距离的重要程度；q 是在[0,1]之间分布的随机数；q_0 是相关参数($0 \leqslant q_0 \leqslant 1$)；$S_{old}$是根据蚂蚁系统中采用的随机比例规则，按概率 $P(i,j)$ 随机选择下一节点 j。概率 $P(i,j)$按下式确定：

$$P(i,j) = \begin{cases} \dfrac{[\tau(i,j)] \cdot [\eta(i,j)]^{\beta}}{\sum\limits_{u \in J(i)} [\tau(i,u)] \cdot [\eta(i,u)]^{\beta}}, & \text{若 } S_{old} \in J(i) \\ 0, & \text{其他} \end{cases}$$

蚂蚁根据状态转移规则 S 确定要经过的下一点，并且重复该过程而形成一条路径，直到找到目标节点 t。蚂蚁每经过一个节点，就按照下式所给出的局部更新规则来更新这个节点的气味。

$$\tau(j) = (1-\rho) \cdot \tau(j) + \rho \cdot \tau_0$$

式中，ρ 是信息素蒸发因子($0 < \rho < 1$)，τ_0 是局部更新常数，取初始信息素的值。在一次迭代中，当所有蚂蚁都完成了自己的路径后，应用全局更新规则，对连接图各节点上的气味进行更新。为了使得搜索过程更直接，这里仅对所有已走过路径中的最短路径上的各节点气味进行更新，其他节点的气味只是进行衰减。全局更新规则如下式所示：

$$\tau(j) = (1-\rho) \cdot \tau(j) + \rho \cdot \Delta\tau(j)$$

式中，$\Delta\tau(j)$为全局更新因子，可由下式确定：

$$\Delta\tau(j) = \begin{cases} (L_{gb})^{-1}, & j \in \text{最短路径} \\ 0, & \text{其他} \end{cases}$$

L_{gb}为所有路径中最短路径长度。

算法的总体结构描述如下：

初始化(连接图的每一个点赋予气味初值构成矩阵$\langle T \rangle_0$)

循环1:(一次迭代)

M 只蚂蚁置于源点 s 等待出发,

第 $j=i+1$ 只蚂蚁出发的条件是第 i 只蚂蚁找到目标 t 或被堵死;

循环2:(一步)

每只蚂蚁根据状态转移规则S选择连接图上的下一点;

再对刚选择的点及时进行局部气味更新。

结束2(所有蚂蚁找到目标 t 点或被堵死)。

进行全局气味更新;

结束1:完成设置的迭代条件。

这里,在最短路径的估计中引入了分形方法,它比一般的线性估计方法更接近于实际的最短路径,所求问题的分形维数从全局的角度反映了该问题的复杂性。由其得出的气味矩阵 T_0 更合理。虽然求其维数需要一定的计算量,但这样能够在迭代次数上有所改进。

从相关实验结果发现,蚁群算法具有很强的全局寻优解的能力,该算法不仅利用了正反馈原理,而且是一种本质并行的算法。但也存在一些缺陷,如搜索时间长。对于大规模布线优化问题,这是一个必须克服的障碍,必须从算法本身着手,进行改善。

6.4 基于蚁群算法的神经网络训练

6.4.1 基本原理

神经网络是当前智能计算和智能控制领域研究的一个热点模型,通过神经网络模型可在未提供相关被控对象和业务模型的情况下达到学习控制的目的。另外,神经网络的并行结构可用硬件实现的方式进行开发,这样就提供了短的、可预知的响应时间。大多数神经网络都使用最陡下降的学习算法进行训练,例如反向传

播算法(BP),这种算法具有收敛时间长、易于陷入局部极值等缺陷。

这里,可将蚁群算法的全局优化和启发式寻优特征用于训练神经网络的权值,以达到神经网络模型的智能寻优目的。其基本思想是:假定网络中有 D 个参数,它包括所有的权值和阈值。首先,对所有这些参数进行排序,对参数 P_i,$1 \leqslant i \leqslant D$,取它的所有可能值组成一个集合 Ω_{P_i}。然后,定义一定数量的蚂蚁智能体,从蚁巢出发去寻找食物。每只蚂蚁从第 1 个集合出发,根据集合中每个元素的信息素状态,随机地从每个集合 Ω_{P_i},$1 \leqslant i \leqslant D$ 中选择一个元素,并且调节相应所选元素的信息素。当蚂蚁在所有集合中完成元素的选择后,它就到达食物源,并且沿着刚走过的路径返回蚁巢,同时调节相应于所选元素的信息素。经过蚂蚁不断迭代,最终可找到参数的最优解。

假定有 M 只蚂蚁,并假定集合 Ω_{P_i},$1 \leqslant i \leqslant D$ 有 N_{P_r} 个元素,$P_j(\Omega_{P_r})$ 表示它的 j 个元素,它的相应信息素用 $P_{P_j}(\Omega_{P_r})$ 表示。在蚁群搜索食物的每一时间步内所有蚂蚁只能选择 1 个元素,并且对不同的蚂蚁,在每一时间步内由它选择的元素应属于不同的集合。此外,I 表示蚂蚁选择的相应元素的信息素增加量,E 表示信息素的挥发,如果 E 小于 0,那么让它恒等于 0。

6.4.2 基于蚁群算法的最优参数搜寻步骤

现在,用蚁群搜寻满足误差限制的最优参数的步骤可描述如下:

步骤 1:初始化所有集合 Ω_{P_r},$1 \leqslant r \leqslant D$ 中的每个元素 $P_j(\Omega_{P_r})$ 的信息素 $P_{P_j}(\Omega_{P_r})$,然后从蚁巢发出 M 只蚂蚁,每只都执行步骤 2。

步骤 2:蚂蚁从第一个集合开始,根据下述规则依次在每个集合中选择一个元素,同时对它的信息素增加 I;

路径选择规则:对集合 Ω_{P_i},蚂蚁根据下述概率随机地选择它

的第 j 个元素：

$$Pro_{P_j(\Omega_{P_i})} = \frac{P_{P_j(\Omega_{P_i})}}{\sum_{l \in N_{P_i}} P_{P_l(\Omega_{P_i})}}$$

步骤3：当所有蚂蚁在每步中完成元素的选择之后，从所有集合的每一元素的信息素量中减去 E；

步骤4：对一只蚂蚁，从所有集合中完成元素的选择，并执行步骤3后，它通过刚走过的路径返回蚁巢，同时根据下述规则增加相应于所选元素的信息素。在此蚂蚁返回蚁巢后，它转向步骤2。此外，在每一时间步，都执行步骤3；

信息素调节规则：

对每只蚂蚁，在它从食物源返回它的蚁巢期间，它根据下式进行信息素的调节。

相应于每个所选元素 P_l 的信息素 P_{Pl}：

$$P_{Pl} = P_{Pl} + \frac{c}{e}$$

式中，c 是一常数，用于调节信息素调节的速度；e 是所有采样值的最大输出误差，定义如下：

$$e = \max_{n=1}^{K} | O_n - O_{ex} |$$

式中，K 是样本数目，O_n 和 O_{ex} 是神经网络的实际输出和期望输出。由此可见，此误差越小，相应信息素的增加就越多。

步骤5：上述步骤一直进行，直到所有蚁群对参数发现了最优解。

6.5 电力系统机组组合问题求解

6.5.1 总体思路

电力系统发电机的机组组合(Unit Commitment，UC)又称为开

停机计划,研究的问题是在保证系统安全运行的前提下以及所研究的周期内,如何合理地安排机组运行,使系统消耗的燃料总量(或支出的费用)最小。这是一个大规模、混合整数、约束非线性组合优化问题。要寻求此问题的绝对最优解,只有通过全部列举所有可能的组合才能得到。因此,对于实际系统,其计算时间是难以接受的。国内外很多学者提出各种方法求解该问题,如启发式方法、动态规划法、整数规划和混合整数规划法、分支定界法、拉格朗日松弛法等。

蚁群算法在求解旅行商(TSP)问题、指派问题(Assignment Problem)、Job-Shop 调度等优化问题中,得到了一系列较好的应用。但是其本身存在一定的不足,如在求解过程中易出现停滞现象,导致算法最终可能收敛于局部最优解。鉴于此,这里介绍的方案中,对蚁群算法的寻优机理进行了研究,针对基本蚁群算法中易出现的停滞现象,从随机优化的角度出发,提出了一种新的转移策略,设计出一种新颖的随机扰动蚁群优化算法(Ant Colony Optimization Algorithm with Random Perturbation Behavior, RPACO),并将其用于求解机组组合问题。同时在求解时,对约束项的处理问题做了进一步程度的探讨。结果表明,RPACO 算法能有效缓解基本蚁群算法的停滞现象,且在获得全局最优解、提高收敛性方面显示了一定的优越性。

6.5.2 最优机组组合问题的数学模型

最优机组组合问题就是寻求一个周期内各个负荷水平下机组的最优组合方式及开停机计划,使相关运行费用最小。目标函数中主要可包括以下两个基本部分:

(1) 机组的发电费用;

(2) 机组的启动费用。

同时,还必须满足一定的约束条件。机组最优投入问题在数学上可以描述为:

$$\min \sum_{t=1}^{T} \sum_{i=1}^{n} [F_i(P_i^t)u_i^t + F_{si}(t)u_i^t(1 - u_i^{t-1})]$$

$$F_i(P_i^t) = a_i + b_i P_i^t + c_i (P_i^t)^2$$

$$F_{si}(t) = \varepsilon_i + \psi_i (1 - e^{q_i^t/\psi_i})$$

这里需满足的等式与不等式约束为：

$$\begin{cases} \sum_{i=1}^{n} P_i^t = P_D^t + P_L^t \\ \dfrac{\sum_{i=1}^{n} u_i^t P_i^{\max} - P_D^t}{P_D^t} \geqslant R^t \\ U_i^t \underline{P_D^t} \leqslant P_i^t \leqslant u_i^t \overline{P_i} \\ (u_i^t - u_i^{t-1})(\overline{\omega}_i^{t-1} - \overline{\omega}_i^{\min}) \leqslant 0 \end{cases}$$

$$\begin{cases} (u_i^t - u_i^{t-1})(q_i^{t-1} - q_i^{\min}) \leqslant 0 \\ P_{i,t+1} < P_{i,t} + \Delta \overline{\omega}_i \qquad (i = 1,2,\cdots,n; t = 1,2,\cdots,T; \\ P_{i,t+1} > P_{i,t} - \Delta \overline{\omega}_i' \qquad l = 1,2,\cdots,N_L) \\ P_l \leqslant P_{l\max} \end{cases}$$

式中，$F_i(P_i^t)$，$F_{si}(t)$分别为第 i 台发电机的发电费用函数及启动费用函数；n，T，N_L 分别为发电机台数、时段数和线路数；u_i^t 为 0～1 变量，$u_i^t=0$ 表示停机，$u_i^t=1$ 表示开机；P_i^t 为第 i 台机组在第 t 时段的出力；P_D^t，P_L^t，R^t 分别为第 t 时段全网负荷、网损、旋转备用率；$\underline{P_i}$，$\overline{P_i}$ 分别为第 i 台机组的最小和最大出力；$\overline{\omega}_i^t = u_i^t(\overline{\omega}_i^{t-1} + 1)$；$\overline{\omega}_i^{t-1}$ 为机组 i 在 $t-1$ 时段已连续运行的时间；$\overline{\omega}_i^{\min}$ 为机组最小启动时间；$q_i^t = (1 - u_i^t)(q_i^{t-1} + 1)$；$q_i^{t-1}$ 为机组 i 在 $t-1$ 时段已连续停机的时间；$q_i^{\min}$ 为机组最小停运时间；$\Delta \overline{\omega}_i$ 为第 i 台机组出力最大增量，即满足机组爬坡速率约束；P_l 为线路有功功率；$P_{l\max}$ 为线路传输功率的限制值。

6.5.3 随机扰动蚁群优化算法

基本蚁群算法中，放到某条路径上的信息素越多且路径越短，则该路径的转移概率越大，同时该路径被蚂蚁选中的概率就越大。这类似于遗传算法中的“轮盘赌法”。鉴于这样一层物理含义，本文设计了如下更为简洁的转移策略：

$$\begin{cases} C_{ij(k)} = \begin{cases} [\tau_{ij}(t)\eta_{ij}]^{\gamma}, j \notin T_{(k)} \\ 0, \text{否则} \end{cases} \\ s \text{ 为 } \max(C_{ij(k)}) \text{所对应的城市} \end{cases}$$

式中，$\gamma > 0$，为扰动因子；$\tau_{ij}(t)\eta_{ij}$的物理含义为单位路径上的信息量。

需要指出的是：上式中的 $C_{ij(k)}$ 不再是“转移概率”，而是路径 ij 的“转移系数”，蚂蚁总是选择转移系数最大的路径，这具有一定的确定性。经分析可以发现，当 γ 取固定值时，该算法与基本蚁群算法相同，仍不可避免出现停滞现象，故有方案提出采用可变的扰动因子，并考虑以下因素：

（1）蚂蚁个体的运动总是沿着转移系数最大的路径移动。当群体规模较大时，由于很难在短时间内从大量杂乱无章的路径中找出一条较好的封闭路径，因此在最初的几次迭代中，为加速算法的收敛，转移系数应取较大的值，才能使得较好路径上的信息量明显高于其他路径上的信息量。

（2）若 γ 一直不变，必将导致某一路径上的信息量远远高于其他路径上的信息量，而此路径并不一定是最优的，这就会导致随后的搜索出现停滞现象。由此，在随后的搜索过程中适当减小 γ 值，一方面可以提高路径选择的多样性，即起到一定的扰动作用，另一方面可以使收敛趋于平缓。

这里，可以采用倒指数关系的曲线来对 γ 进行描述：

$$\gamma = ab^{b/k}$$

式中，$k = 1, 2, \cdots, M$；M 为最大迭代次数；a, b 为扰动尺度因

子，$a>0$，$b>0$。由上式可知，随着迭代次数的增大，γ 值最终趋近于 a，而 b 的大小决定了曲线趋近于 a 的快慢程度。

考虑到基本蚁群算法中易出现的停滞现象，可以设计一个随机选择策略，并在进化过程中动态地调整随机选择的概率。最终，可以设计如下的具有随机扰动特性的转移策略：

$$\begin{cases} C_{ij(k)}(t) = \begin{cases} [\tau_{ij}(t)]^{\alpha}\eta_{ij}, & U \leqslant P_m, j \notin T_{(k)} \\ [\tau_{ij}(t)\eta_{ij}]^{\gamma}, & U > P_m, j \notin T_{(k)} \\ 0, \text{其他} \end{cases} \\ s \text{ 为 } \max(C_{ij(k)}) \text{所对应的城市} \end{cases}$$

式中：γ 为具有倒指数的扰动因子；$P_m \in (0,1)$，称为随机变异率；U 是(0,1)中均匀分布的随机数。

上式将确定性选择与随机选择相结合。确定性选择引导蚂蚁朝着最优方向搜索，随机选择导致计算转移系数时具有一定的随机性，从而有效地避免搜索陷入局部最优解。正是两者的共同作用才使随机扰动蚁群优化算法具有更强的全局搜索能力。

6.5.4 参数选取

在随机扰动蚁群优化算法中，α,ρ,Q,P_m,a,b 是非常重要的参数，其选取对于计算结果影响较大。对于上述参数，目前解析法还难以确定其最佳组合。这里，有关参考文献中通过大量的数字仿真来进行参数的优化设置。例如，以 gr24(24 个城市，276 条线路)、bayes29(29 个城市，406 条线路)、gr48(48 个城市，1128 条线路)的 TSP 问题为例，对随机扰动蚁群优化算法中的参数进行了分析，总结出了参数选取方法，并进一步确定了某些参数的最佳取值范围。

一般说来，蚁群算法的参数可通过反复试凑得到，显然这对算法的计算效率和收敛性将产生不利影响。为此，通过大量的数字仿真，可总结出一种较有效的参数选取方法。具体步骤如下：

(1) 大量实验发现:参数 α 和 Q 的取值范围较大;而参数 ρ 和 P_m 取值相对固定(在 0 ~ 1 之间)。因此,首先可随机设定一组 ρ 和 P_m 的值,来调整 α 和 Q,得到较理想的解,称之为"粗调"。

(2) 在基本确定 α 和 Q 值之后,反过来调整 ρ 和 P_m 寻找更优的解,称之为"细调"。此后,在"细调"得到 ρ 和 P_m 的基础上,再进行"粗调",得到更好的 α 和 Q 值,如此反复,最终确定出一组较为理想的参数组合。

这种方法是在对不同 TSP 问题进行大量数值仿真的基础上总结出来的,具有一定的普遍意义。

应用前述方法,可以初步确定以下参数的取值范围,当参数在此范围内取值时可得到较好的解:

(1) 参数 α 表示某一路径的信息量对蚂蚁选择路径的影响程度,参数 Q 的大小决定路径上信息量的更新程度。有关数值仿真可知:α 的取值在 0.1 ~ 0.5 之间;Q 的取值范围为 50 ~ 100 时可得到较好的解。

(2) 变量 $\rho \cdot \tau$ 的物理含义为残留的信息量,需要忘记一部分过去积累的信息,以便更好地利用最新的信息。所以,若 ρ 取值过小,则不能很好地利用过去积累的信息,若 ρ 很大,则达不到信息有效更新的目的。经大量数字仿真发现:当 $\rho < 0.1$ 时,ρ 就会失去调节作用;当 $\rho > 0.9$ 时,将导致收敛于较差解。这里建议 ρ 的取值范围为 0.5 ~ 0.9。

(3) 扰动尺度因子 a 的取值对算法的性能影响不大,b 的取值范围为 0.5 ~ 5。这里推荐以下几组 (a,b) 较优的组合:(0.5,2),(1,2),(2,4)。

6.5.5 随机扰动蚁群优化算法在机组最优组合中的应用

6.5.5.1 机组组合问题的动态模型

首先,需要定义以下概念:

状态:一些机组投入运行,而另一些机组不投入运行所形成的

机组组合构成一个状态。

决策:从某时段(小时)机组组合的某状态到下时段机组组合的另一状态形成一个决策。

路径:从时段1到时段T的所有时段中,每一时段任取一种机组组合状态,这样组成的一个决策集合称为一条路径。

根据上述概念,机组最优组合问题模型可以转化为一个动态模型,表示从$t-1$时段的状态k到t时段的状态l的最小运行费用。即:

$$\begin{cases} F_t(U_t^l) = \min\{\psi_t^l(U_{t-1}^k, U_t^l)\}, t \in T \\ s.t., \text{等式和不等式约束} \end{cases}$$

式中,

$$\psi_t^l(U_{t-1}^k, U_t^l) = \sum_{i \in G} F_i(P_i^t) + \sum_{i \in G} F_{s_i}(q_i^{t-1}) + C_P(k,l) + F_{t-1}(U_{t-1}^k)$$

上式中第一项表示t时段的发电费用,第二项表示发电机的启动费用,第三项表示违反约束时的惩罚费用,最后一项表示从时段1到时段$t-1$运行成本的累积。

建立了机组最优组合问题的动态模型后,该问题可被看做是一个多阶段搜索问题,利用前述概念可将该模型转化成类似TSP问题的模式:

$$\min\left[\sum_{i=1}^{n-1} c_t(s_{\pi(i)}, s_{\pi(i+1)}) + c_t(s_{\pi(n)}, s_{\pi(1)})\right]$$

式中,$c_t(s_i, s_j)$表示从状态i到状态j的转移费用(包括机组的启动费用及惩罚费用,若费用为0,则在计算过程中取一个较小的值),对应于TSP问题中的路径长度,即两城市之间的距离,需要说明的是只有在Tabu表中某状态的标志为1时,才可获得相应路径的长度。由于机组的启动费用是机组停机时间的函数,因此路径及其长度在计算过程中是动态变化的。$\pi(i)$表示第i时段所

有可选状态集合。

这样,机组最优投入问题就可以像 TSP 问题一样,利用随机扰动蚁群优化算法来求解。

6.5.5.2 等式和不等式约束的处理

机组最优组合问题为约束非线性规划问题,而随机扰动蚁群优化算法属于无约束优化方法。这里可采用三种方法来处理各种约束。

(1) 经济调度:等式约束(功率平衡方程)、机组输出功率上下限约束以及机组功率爬升速度约束(Ramp Rate),可在经济调度时进行考虑。

进行功率平衡时,若计及网损的影响,采用直流潮流 B 系数法计算网损,可直接建立网损与发电机有功之间的关系,即该系统的 B 系数:B_L,B_{l0},B_0。进行经济调度时,采用协调方程式法分配各机组输出功率,以满足功率平衡约束。若不计网损,则按等耗量/费用微增率法进行调度。

对于机组输出功率上下限约束,在经济调度过程中当一台机组输出功率越限(上/下)时,将其输出功率定在限值(上/下)处,然后对剩余机组重新调度,直到所有机组均满足输出功率上下限约束并保持功率平衡。

对于机组爬坡速率约束,如果已知机组 t 时段的输出功率 P_i^t,在 $t+1$ 时段进行经济调度时,若 $P_i^{t+1} > P_i^t + \Delta\overline{\omega_i}$($\Delta\overline{\omega_i}$为机组最大爬升功率),则令 $P_i^{t+1} = P_i^t + \Delta\overline{\omega_i}$,作为机组输出功率上限;若 $P_i^{t+1} < P_i^t - \Delta\overline{\omega_i}$,则令 $P_i^{t+1} = P_i^t - \Delta\overline{\omega_i}$,作为机组输出功率下限,然后按处理功率越限的办法处理机组调整速度约束。

(2) Tabu 表限制:旋转备用、机组最小启动/停机时间约束在形成蚂蚁在各时段的 Tabu 表时予以考虑,对满足上述约束的状态在 Tabu 表中将其置 1,否则置 0。

需要说明的是:这里的 Tabu 表与基本蚁群算法中的表有所不

同,该表对应于时段 t,记录的是该时段所有允许转移的状态。

(3) 构造增广目标函数:对于线路 N 安全性约束,可以惩罚方式引入目标函数中,构造如下增广目标函数:

$$\min \sum_{t=1}^{T} \sum_{i=1}^{n} [F_i(P_i^t)u_i^t + F_{si}(t)\gamma_i^t(1 - u_i^{t-1})] + C_E\{\max[0, \sum_{k \in N_L}(T_l^k - T_{l\max}^k)]\}$$

式中,C_E 为惩罚因子,为一个较大的常数。

计算线路传输功率时,可将线路有功表示为发电机有功的线性关系。这样,当发电机运行状态或输出功率发生改变时,相应的线路有功功率可方便地求解出来,而不必再计算潮流。

6.5.5.3 *算法流程图*

利用随机扰动蚁群优化算法求解电力系统机组组合问题的算法流程见图 6-4。

以上通过提出采用倒指数曲线描述的扰动因子,并且设计随机选择策略和扰动策略,构造了一种新颖的随机扰动蚁群优化算法,并将其用于求解机组组合问题。求解时,对模型的转化、约束项的处理做了进一步的探讨。所定义算法能有效缓解基本蚁群算法的停滞现象,并且在获得全局最优解方面显示了一定的优越性。

6.6 基于蚁群算法的多播路由算法

蚁群算法具有收敛速度慢,易限于局部最小点等缺陷,通过对基本蚁群算法进行改进,在每次循环结束时,保留最优解,自适应地改变挥发度系数,引入遗传算法的交叉算子,提出一种基于基本蚁群算法的有时延约束的多播路由算法模型。仿真结果表明,基于改进基本蚁群算法的多播路由算法模型可以稳定地获得优于现有启发式算法的解,是一种有效的多播路由算法,该算法也适用于并行执行和应用。

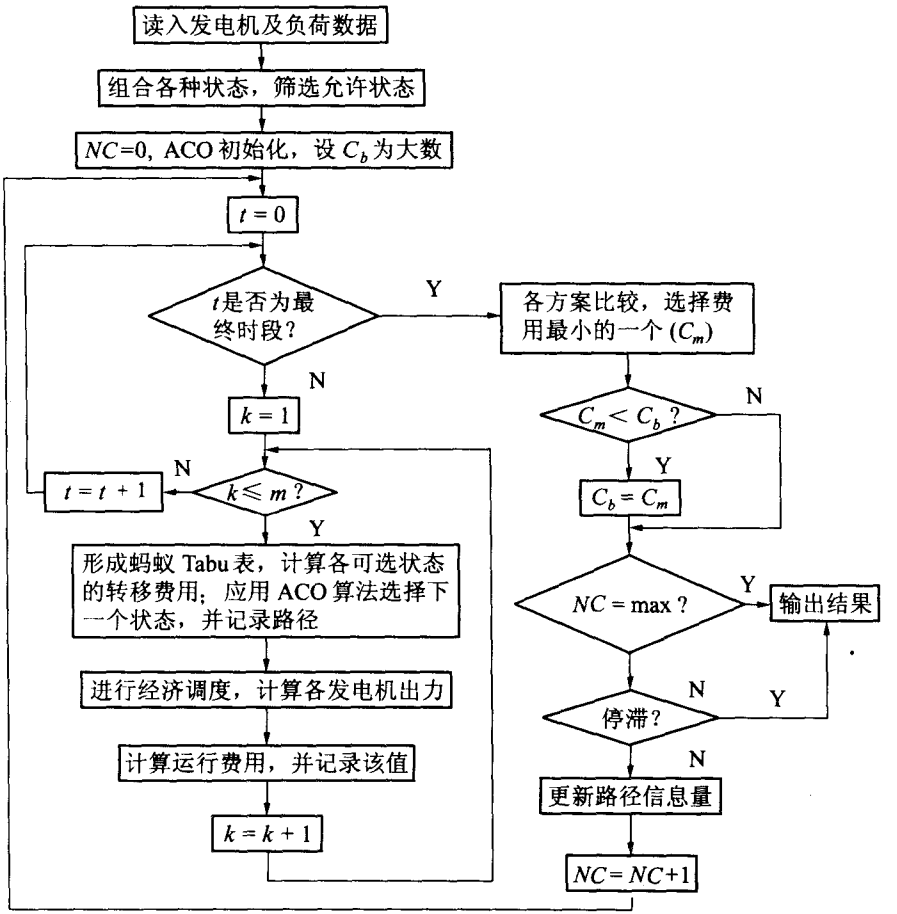


图 6－4 算法流程图

蚁群算法是一种随机搜索算法，与其他模拟进化算法一样，通过由候选解组成的群体的进化过程来寻求最优解。在通信网络中，基本蚁群算法可用于解决容量平衡问题。

多播通信路由问题的实质就是寻找连接一组点的基于某种代价的最小连接树，即考虑的是有时延约束的最小代价树问题。该问题属于 NP 完全问题，一般地说，最优算法无法在多项式时间内完成，而启发式算法虽难度降低，但只能得到次优解。下面用改进的基本蚁群算法来解决多播路由问题，并提出解决这一问题的新思路。

6.6.1 算法模型

本节为每一对源、目标节点建立一个路由表，路由表中存储源、目标节点间满足约束条件的 lk 个路由，路由表的建立可以通过 K-shortest-path 算法完成。每一对节点间的路由可记为 $i_d(i=1,2,\cdots,lk;d\in D)$。其中，$i$ 为路由的序号，下标 d 为目标节点。每只蚂蚁为每个目标节点选择一个路由，组合为一个满足约束条件的树。

设 $\tau_{id}(t)$ 为时间 t 蚂蚁留在路由 i_d 上的信息量，每只蚂蚁在时间 t 开始一次新的循环，每次循环蚂蚁为所有目标节点都选择一个路由。一次循环结束时时间更新为 $t+n$，蚂蚁会根据下式更新路由 i_d 上的信息量：

$$\tau_{id}(t+n)=\rho\tau_{id}(t)+\Delta\tau_{id}(t)$$

式中，$0<\rho<1$ 为常数；$1-\rho$ 为 $\tau_{id}(t)$ 在时间 t 和 $t+n$ 之间的挥发程度。

$$\Delta\tau_{id}(t)=\sum_{ak=1}^{m}\Delta\tau_{id}^{ak}(t)$$

m 为蚂蚁的总数；$\Delta\tau_{id}^{ak}(t)$ 为在 t 和 $t+n$ 间，由第 ak 只蚂蚁引起的路由 i_d 上信息量的变化。

$$\Delta\tau_{id}^{ak}=\begin{cases}Q/L_{ak}, & \text{第 } ak \text{ 只蚂蚁选择 } i_d(t,t+n)\\ 0, & \text{其他}\end{cases}$$

式中，Q 为常数；L_{ak} 为蚂蚁 ak 求得的树 T 的代价和。蚂蚁 ak 以概率 $P_{id}^{ak}(t)$ 选择路由 i_d 作为本次循环 i 到 $d\in D$ 的通路。

$$P_{id}^{ak}(t) = \frac{[\tau_{id}(t)]^{\alpha}[\eta_{id}]^{\beta}}{\sum_{j=1}^{lk}[\tau_{jd}(t)]^{\alpha}[\eta_{jd}]^{\beta}}$$

式中，η_{id} 为启发式信息，这里取 $\eta_{id}=1/d_{id}$，d_{id} 为路由 i_d 的长度；α,β 为常系数，分别表示 $\tau_{id}(t)$ 和 η_{id} 的重要程度。这样每只蚂蚁为所有目标节点都选择了一个路由，将所有目标节点的路由综合，然后去掉重合的边，就可以得到一个连接所有源、目标节点的树，这棵树同时满足时延约束。

6.6.2 算法的改进

为了提高基本蚁群算法的全局搜索能力和搜索速度，可将基本蚁群算法作如下改进：

（1）保留最优解。在每次循环结束后，求出最优解，将其保留。

（2）当问题规模比较大时，由于信息量挥发系数 ρ 的存在，使那些从未被搜索到的解上信息量会减小到接近于0，降低了算法的全局搜索能力。通过减小 ρ 虽然可以提高算法的全局搜索能力，但又会使算法的收敛速度降低。因此，这里可自适应地改变 ρ 值。ρ 的初始值 $\rho(t_0)=1$；当算法求得的最优值在 M 次循环内没有明显改进时，ρ 可减为：

$$\rho(t) = \begin{cases} \lambda\rho(t-1), & \text{若 } \lambda\rho(t-1) \geqslant \rho_{\min} \\ \rho_{\min}, & \text{其他} \end{cases}$$

式中，$\rho_{\min}$ 可以防止 ρ 过小以降低算法的收敛速度；λ 为约束系数，$0<\lambda<1$。

（3）在算法运行的初始阶段，每个路由上的信息量相差不大，通过信息的正反馈，较好解的信息量增大，从而逐渐收敛。当问题规模比较大时，这一过程需要较长的时间。为了提高算法的收敛速度，引入遗传算法的交叉算子，在每次循环结束，随机选择两只蚂蚁，将其解进行交叉操作，交叉点随机产生。如果交叉所得的解

优于原来的解，则用新的解代替原来的解，否则抛弃新的解。由于每只蚂蚁对每个目标节点的路由独立操作，交叉操作将会引入路由的不同组合，不会引入不可行解。

算法的具体步骤如下（其中 NC 为循环次数，n 为目标节点数）：

```
begin
    初始化
    η_id = 1/d_id; t = 0; NC = 1; τ_id = 0; Δτ_id = 0
    while (未满足终止条件);
        {for (ak = 1; ak ≤ m - 1; ak + +)
        {将 m 只蚂蚁放置于源节点 r 上}
        for (i = 1; i < n; i + +)
            {for (ak = 1; ak ≤ m - 1; ak + +)
                {蚂蚁 ak 以概率 P_id^ak 为第 i 个目标节点选
                择路由}}
            for (ak = 1; ak ≤ m; ak + +)
            {随机选择一个除 ak 以外的蚂蚁，在随机选择
            的交叉点上与蚂蚁 ak 的结果进行交叉操作
            if 交叉产生路由树优于 ak 的结果
            {将交叉产生的最优解赋于 ak}}
            求出最佳结果，将最佳结果赋于第 m 只蚂蚁
            if 最佳解 = M 个循环以前的最佳解
            {更新 ρ}
            更新 Δτ_id; 更新 τ_id; Δτ_id = 0; NC = NC + 1;}
        输出最佳结果
end
```

以上描述的一种基于基本蚁群算法的有时延约束的多播路由算法模型中，将基本蚁群算法作了一定的改进，是一种有效的、具有

实际意义的多播路由算法,同时该算法也适用于并行执行和应用。

6.7 蚁群算法在数据挖掘中的应用

6.7.1 背景介绍

数据挖掘(或知识发现)就是从大量的数据中抽取以前未知、并具有潜在可用特征的模式,以此来帮助人们理解大量的原始数据。也就是说,数据挖掘是从大量数据中发现正确的、新颖的、潜在有用的,并能够被理解的知识或规则的过程。它是一个多科性的领域,核心是机器学习、统计学和数据库的结合。这里,必须强调在数据挖掘中所提取的知识或规则不仅应该准确,而且对于用户来说也应是可以被理解的。当我们把数据挖掘用于决策支持系统时,知识或规则易于理解的特性是相当重要的。毕竟,如果获取的知识或规则不能被用户理解,那么也就无法被解释和验证。在这种情况下,用户也不可能充分信任提取的知识或规则,并将其用于决策的制定。

目前,对数据挖掘的研究和开发表明,数据挖掘需要覆盖各种各样不同的应用任务,包括从数据的预处理到关联规则、聚类分析、数据分类、偏差检查、序列模式等特定模式,其中每一种功能可以被认为是通过数据挖掘算法解决的一类问题。因此,应该设计一种数据挖掘算法是必须解决的首要任务。

这里,主要讨论数据挖掘算法来实现数据分类的功能。

在数据挖掘的分类功能中,获取的知识通常可表述为 IF-THEN 的模式:

IF < conditions > THEN < class >

该模式(规则)的条件部分(IF 部分)包括了一个条件集合,通常可以用逻辑连接符(AND)进行连接。这里所指的每一个规则就是一个条件字段(元素),所以规则的条件部分就是由很多条件元素进行的一个逻辑连接,有如下形式:IF term1 AND term2

AND…，其中，每一个元素(term)又是一个三元组 < 特征属性，操作符，特征值 > ，比如 < gender = female > 。

模式(规则)的推理部分(THEN 部分)是对满足规则条件部分中所有条件的数据进行的类别分配。从数据挖掘的观点来看，在待发现的规则数目和部分规则条件中，在元素的数目不是很多的情况下，该种数据表示方式具有易于用户直观理解的优点。下面我们将尝试将蚁群算法用于数据挖掘技术中数据的分类规则提取。

6.7.2 用于数据挖掘的蚁群算法

这里详细讨论所提出的用于数据挖掘中数据分类规则的蚁群算法，称为挖掘蚂蚁(Ant Miner)。主要通过下面四部分来加以描述：挖掘蚂蚁概述、启发函数、规则修改和信息素更新。

6.7.2.1 挖掘蚂蚁概述

在蚁群算法中，每一只蚂蚁构成了问题的一个解。这里，问题的目标是发现分类规则。每一个分类规则应具有如下形式：

IF < term1 AND term2 AND… > THEN < class >

每一个条件元素是一个三元组 < 特征属性，操作符，特征值 > ，这里的特征值属于特征空间。三元组中的操作符元素是一个关系运算符。当前的数据挖掘蚂蚁(Ant Miner)仅具有分类特征，所以三元组中操作符元素通常为“ = ”，连续值(实数值)的特性应首先在前道步骤中进行离散化。

以下具体描述了蚂蚁挖掘算法。

在算法开始的时候，挖掘规则列表为空，而训练集合包括所有的训练情况。算法的每一个 WHILE 循环中的一定数量的 REPEAT-UNIL 循环执行过程，发现一个分类规则。该规则被添加到挖掘规则列表中，符合规则的训练实例(实例满足规则的条件部分，并且具有规则推理部分的类别推测)被移出训练集合。这一过程在不满足条件的训练实例数大于用户定义门限值(称为

Max Uncovered Cased)的情况下重复执行。

算法中 REPEAT-UNIL 循环的每一次迭代包含三个步骤:规则构建、规则修改和信息素更新。下面对此进行详细描述。

首先,Ant_k开始于一个空规则集合,即一个条件为空的规则。并且,每隔一定的时间间隔在当前偏好规则集合中添加一个元素。每一只蚂蚁针对当前偏好路径的集合中所经过的路径构造一个规则。相似地,对当前偏好规则中增加元素的选择相应于对当前路径扩展方向的选择。对当前偏好规则的添加元素选择,不但取决于与问题相关的启发函数(η),还取决于对应于每一个规则元素的信息素(τ)数量。Ant_k 每隔一定时间就在当前的偏好规则集合中添加一个元素,直到以下两点满足判断停止条件。

(1) 被添加到规则集合中的任意元素,能够使得规则中包含一定数量的实例,该数量应该小于被称为满足每个规则的实例数目最小值(Min cased per rule)的用户定义的门限值。

(2) 所有的特征都已经被蚂蚁使用过,因此没有更多的特征能够被添加到条件集合中。注意,为了避免无效的规则,如"IF (Sex = male) AND (Sex = female)",每一个特征在每条规则中仅能出现一次。

算法:蚂蚁挖掘算法的一种高层描述

TrainingSet = {all training cases}

DiscoveredRuleList = [ ]; /* 用空集合初始化规则列表 */

WHILE(TrainingSet > Max uncovered cases)

$t = 1$;

$j = 1$;

用相同数量的信息素初始化所有特征;

REPEAT

Ant_t 由一个空规则集合开始,每隔一定的时间,在当前的规则集合中添加一条规则,同时动态构造出一个分类规则 R_t;

修改规则 R_t；

对所有痕迹更新信息素：在 Ant_t 所经过的路径上增加信息素（正比于 R_t），在其他路径上减少信息素（模拟信息素挥发）

IF（R_t 等于 R_{t-1}）　　/*更新收敛检验*/

　　THEN $j=j+1$；

ELSE $j=1$；

END IF

$t=t+1$；

UNTIL（$t \geqslant No\ of\ ants$）OR（$j \geqslant No\ rules\ converg$）

在所有的蚂蚁构造的规则中 R_t 选择最好的规则 R_{best}；

增加规则 R_{best} 到 DiscoveredRuleList；

TrainingSet = TrainingSet | set of cases correctly covered by R_{best} |；

END WHILE

第二，为了去除不相关的规则元素，应对蚂蚁 Ant_t 构造的规则集合 R_t 进行修改。到目前为止，所指的这些不相关元素，可能已经被包含进规则集合中，这是因为在规则选择过程中出现的随机变化特性和（或）在选择过程中使用了局部的启发函数，该函数每次仅考虑一个特征的影响，而忽略特征间的相互作用。

第三，更新每段路径上的信息素数量：增加 Ant_t 所访问路径上的信息素（按照规则 R_t），同时减少其他路径上的信息素（模拟信息素的挥发）。接下来，其他蚂蚁开始构造各自的规则，使用更新后的信息素来引导它们的搜索过程。这一过程重复执行，直到满足下面的两个条件。

（1）构造规则的数目等于或大于用户规定的门限值No _ of _ ants。

（2）当前蚂蚁 Ant_t 构造的规则与前 No _ rules _ converg − 1 只蚂蚁构造的规则完全相同，这里的 No _ rules _ converg 代表测试蚂

蚁收敛性时使用的规则数目。

一旦 REPEAT-UNTIL 内的循环结束，则选择所有蚂蚁构造规则中最好的规则，并将其添加到发现规则的列表中。正如前面所提到的，系统开始 WHILE 循环中一次新的迭代，并重新为每条路径初始化相同数量的信息素。

在蚁群算法的标准定义中，群体规模代表在两次信息素更新之间构造路径的蚂蚁数。按照这种定义，在 WHILE 循环的每一个迭代中，Ant Miner 由一只蚂蚁构成种群，这是因为，信息素是在一只蚂蚁构造一个规则之后更新的。因此，严格地说，Ant Miner 中 WHILE 循环的每一次迭代只有一只蚂蚁，该蚂蚁用来完成很多次迭代。注意，WHILE 循环中不同的迭代相应有不同的群体规模，因为不同群体的蚂蚁可用于解决不同的问题，即不同的训练集合。然而，为了简化算法的描述，这里我们将该独立的蚂蚁进行的第 t 次迭代称为第 t 只蚂蚁（Ant_t）。

从数据挖掘的观点来看，Ant Miner 运行操作的核心是算法中 REPEAT-UNTIL 循环的第一步，其中当前蚂蚁反复在它的当前偏好规则集合中添加元素。$term_{ij}$ 满足 $A_i = V_{ij}$ 的形式，这里的 A_i 代表第 i 个特征，V_{ij} 代表中 A_i 中的第 j 个值。$term_{ij}$ 被选择添加到当前偏好规则集合的概率为：

$$P_{ij} = \frac{\eta_{ij} \cdot \tau_{ij}(t)}{\sum_{i=1}^{a} x_i \cdot \sum_{j=1}^{b_i} [\eta_{ij} \cdot \tau_{ij}(t)]}$$

式中，η_{ij} 为 $term_{ij}$ 与问题相关的启发函数值。η_{ij} 值越大，表明 $term_{ij}$ 对分类相关性越大，被选择的概率也就越大。

$\tau_{ij}(t)$ 表示在循环 t 与 $term_{ij}$ 相关的信息素数量，对应于当前蚂蚁通过的 i 和 j 之间的路径上当前有效信息素的数量。蚂蚁构造的规则越好，其所访问路段上增加的信息素就越多。因此，随着时间的进行，产生最好的路径，即添加到规则集合中的最优元素

(特征—特征值对),将具有越来越多的信息素,它们被选择的概率也将随之增加。

a:特征的总数目

x_i:若特征 A_i 未被当前蚂蚁使用,则设置为1,否则,设为0

b_i:第 i 个特征属性中特征值的数目

按照上面公式计算的概率和下面的两条限制条件选择 $term_{ij}$,并将其添加到当前规则集合中。

(1) 特征 A_i 未被包含在当前偏好规则中。为了满足这条限制条件,蚂蚁必须"记忆"哪些元素(特征—特征值对)包含在当前偏好规则中。

(2) 如果元素 $term_{ij}$ 使规则集合包含的实例少于预先定义的最小值(称为 Min _ cases _ per _ rule 门限值),那么,$term_{ij}$ 就不能被添加到当前偏好规则集合中。

一旦规则条件判断完成,系统就开始选择可以使规则产生最佳效应的规则推理(即推理类别)。

6.7.2.2 *启发函数*

对于可以被添加到当前规则集合中的每一个 $term_{ij}$,Ant Miner 均计算启发函数 η_{ij} 的值。这是对该元素质量的一个估计,而元素质量是指元素所具有的提高规则推理准确性的能力。启发函数是基于信息理论的。更准确地说,对于 $term_{ij}$ 的 η_{ij} 值包含元素熵的测量。其中每一个均满足 $A_i = V_{ij}$(这里的 A_i 代表第 i 个特征,V_{ij} 代表中 A_i 中的第 j 个值)的 $term_{ij}$,其熵是:

$$H(W \mid A_i = V_{ij}) = -\sum_{w=1}^{k} (P(w \mid A_i = V_{ij}) \cdot \log_2 P(w \mid A_i = V_{ij}))$$

式中,W 是指类的特征(即其范围包含被推理类的特征),k 是类的数目,$P(w \mid A_i = V_{ij})$ 是观察类 w 满足条件 $A_i = V_{ij}$ 的经验概率。

$H(W|A_i = V_{ij})$值越大,类就越可能被均匀分布,因此当前蚂蚁选择将$term_{ij}$添加到其偏好规则中的概率值越小。为了便于在决定当前偏好规则集合的概率式中使用,有必要对启发函数值进行规范化。为了进行规范化操作,要求$H(W|A_i = V_{ij})$在$0 \leqslant H(W|A_i = V_{ij}) \leqslant \log_2 k$的范围内变化,其中$k$是实用类的数目。因此,规范化的信息理论启发函数可表达为:

$$\eta_{ij} = \frac{\log_2 k - H(W|A_i = V_{ij})}{\sum_{i=1}^{a} x_i \cdot \sum_{j=1}^{b_i} [\log_2 k - H(W|A_i = V_{ij})]}$$

式中,a,x_i和b_i与决定当前偏好规则集合公式中的相关符号有相同的含义。

注意,在一般情况下,$term_{ij}$的$H(W|A_i = V_{ij})$是相同的,并且可忽略元素所在规则集合的内容。因此,为了节省计算时间,所有$term_{ij}$的$H(W|A_i = V_{ij})$可在一个预先进行的程序中进行计算。

在上面的启发式函数中,还应该注意两点:首先,如果特征A_i的V_{ij}值未在训练集合中出现,那么$H(W|A_i = V_{ij})$应被设置为它的最大值$\log_2 k$,这相当于赋予$term_{ij}$最小的推理能力。第二,如果所有的实例属于相同的集合,那么$H(W|A_i = V_{ij})$可设置为0,这相应于赋予$term_{ij}$最大的推理能力。

用于 Ant Miner 进行熵测量的启发式函数,与用于决策树算法的启发函数应属于同一种类型。决策树与 Ant Miner 之间关于启发函数的主要不同是:在决策树中,熵被作为整体的一个特征进行计算。因为,对树的扩展,需要选择一个整体特征。而在 Ant Miner 中,仅对一个特征—特征值对的熵进行计算,因为,对规则的扩展选择的是特征—特征值对。

另外,这里需要强调,在传统的决策树算法中,熵的测量仅在树的构建过程中使用,而在 Ant Miner 算法中,熵的测量出现在信息素更新过程中。这就使得 Ant Miner 的规则构建过程更具鲁棒

性,并且在搜索过程中不易陷入局部最优。这是因为,信息素更新所提供的反馈信息更有助于纠正熵测量产生的错误。这里,熵的测量是一个局部启发测量,包括即时的特征,它对特征间能够进行相互作用的情况相当敏感,因为信息素更新直接根据的是整体规则的性能。

在 REPEAT-UNTIL 循环的开始阶段,所有元素具有相同的信息素时,Ant Miner 进行规则构建的过程应该导向较差的规则。

6.7.2.3 规则修改

规则修改是数据挖掘领域的通用技术。规则修改的主要目标是去除无关的元素,这样才有助于避免训练数据的溢出。规则修改的另一个目的是简化规则,因为较为简洁的规则相对于较长的规则,更易于用户的理解。

在当前蚂蚁完成规则构建时,规则修改程序就可加以调用。

其基本思想是:在能够提高规则质量的情况下,每隔一段时间去除一个元素,并反复进行如此的操作。更准确地说,在第一次迭代中,规则集是满的,而在接下来的迭代中,则按顺序去除规则集合中的每一个元素,并且用一个规则质量函数来计算最后得到规则的质量。这一过程可能包括重置规则序列中的类,因为需要进行修改的实例中,其大多数类可能与满足原始规则实例中的大多数类不同。这一过程可重复执行,直到规则中只有一个元素或者没有元素为止,元素的去除有助于提高规则集合的质量。

6.7.2.4 信息素更新

每一个 $term_{ij}$ 对应于蚂蚁所经过路径中的一个路段。在算法中的 WHILE 循环中,每一次迭代被初始化为相同数量的信息素。因此,当第一只蚂蚁开始搜索时,所有的路径具有相同的信息素量。

这时,在每一个路径位置上遗留的信息素初始量与所有特征值的数量成反比,可按如下定义:

$$\tau_{ij}(t=0)=\frac{1}{\sum_{i=1}^{a} b_i}$$

式中，a 是特征的总数量，b_i 是能够被特征 A_i 采用的可能值数量。

规范化该函数的返回值，以方便在公式计算中使用，并将该值与启发函数结合起来。一旦蚂蚁构建其规则，规则就被修改（参见算法），路径上所有段上信息素量都必须被更新。该信息素更新满足以下两个基本观点：

（1）与蚂蚁找到的规则（修改后的规则）中的每一个 $term_{ij}$ 相关的信息素数量与规则的质量呈比例增加；

（2）与未出现在规则中的每一个 $term_{ij}$ 相关的信息素数量减少，模拟了自然界蚁群中信息素的挥发。

在有用元素信息素需要增加时，应增加与每一个蚂蚁所发现规则中出现的与每一个 $term_{ij}$ 相关的信息素量，就相当于增加蚂蚁所访问路径上的信息素。在规则发现的过程中，这相当于增加 $term_{ij}$ 未来被其他蚂蚁选择的概率，也相当于成比例地增加了规则质量。如果将规则质量记作 Q，由公式 $Q=\text{sensitivity}\cdot\text{specificity}$ 来加以计算，可以定义为：

$$Q=\frac{TP}{TP+FN}\cdot\frac{TN}{FP+TN}$$

式中，

TP——真的正值，规则集合中的实例数，该规则集合包含能够由规则推理所得到的类；

FP——假的正值，规则集合中的实例数，该规则集合包含一个不同于由规则推理得到的类所组成的类；

FN——假的负值，规则集合之外的实例数，但该规则集合包含能够由规则推理得到的类；

TN——真的负值，规则集合之外的实例数，该规则集合包含一个不同于由规则推理得到的类所组成的类。

Q 的值被限制在范围 $0 \leqslant Q \leqslant 1$ 内，并且 Q 值越大，规则质量越高。$term_{ij}$ 的信息素更新可根据下式进行：

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \tau_{ij}(t) \cdot Q, \forall i,j \in R$$

式中，R 是出现在蚂蚁第 t 次迭代所构造规则中的元素集合。

因此，对于所有出现在当前蚂蚁已发现规则中的 $term_{ij}$，应增加其信息素的数量，增加的数值为来自于 Q 的部分信息素当前值的一部分。

在减少未被使用元素的信息素时，正如上面提到的，与未在当前蚂蚁发现规则中出现的每一个 $term_{ij}$ 相关的信息素量必须被减少，这是为了模拟真实蚁群中信息素的挥发。在 Ant Miner 中，信息素的挥发是按照一个间接的方式进行的。更准确地说，是通过规范化信息素 τ_{ij} 来实现未使用元素信息素挥发的影响。规范化过程是将每一个 τ_{ij} 值除以所有 $\tau_{ij}(\forall i,j)$ 的和。这里，可使用增加信息素的方法来进行信息素的挥发。因此，在规范化时，未被使用元素的信息素的量可通过将它的当前值除以所有元素的信息素之和来加以计算，其最终的影响将会使每一个未被使用的元素规范化的信息素量减少。

第7章　蚁群算法拓展及应用

7.1　引言

从本书前面章节的论述，可以知道，蚁群算法在离散空间中的寻优能力十分突出，这已为广大研究人员所关注的。而作为智能自动化领域的研究人员，作者自然会去关心其在连续空间中的应用前景，并且关心其在系统辨识和智能控制领域的应用前景。经过多年来的努力，课题组在国家自然科学基金等科研项目支持下，在此领域做了一定的工作。本章所要论述的正是连续空间内的寻优问题。如何在连续空间内进行合理的蚁群算法定义，是本章的论述重点。连续空间寻优中的应用是本章的主题。

本章中，在对传统蚁群算法理论作出概括性总体论述之后，针对连续空间寻优问题，将蚁群算法在连续空间进行扩展，定义蚁群合理的信息量分布函数和迁移规则，并给出具体的寻优算法。为证明本章所给算法的有效性，在实例研究中，将其用于一维空间的多极值函数寻优，以及非线性连续函数的寻优问题求解。在多维连续空间的寻优中，将其用于线性系统的参数辨识。仿真计算的结果证明，蚁群总体都能趋于连续函数的最优解和线性系统的准确参数。在本章最后，针对连续空间的寻优问题，将蚁群算法的适用特征做了总结，并对今后的工作进行了展望。

7.2　用于连续空间寻优的蚁群算法

7.2.1　用于离散空间寻优的蚁群算法概述

现在，让我们先从另一个角度对用于离散空间寻优的蚁群算法来加以理解。蚁群算法对在离散空间内的寻优问题求解中，问

题各分量的不同组合对应于多维离散空间内的各个点，其中每个点的每一维分量对应于所求解优化问题的各个分量，而每个点又与所求解问题的不同解答相对应。离散空间的寻优问题的目标就在于在给定点集中设定相应的搜索算法，以使与问题最优解相对应的点（或点集）以递增的概率被选中，并最终收敛于与问题最优解相对应的点（或点集）。

以典型的离散空间组合优化问题——TSP 问题为例。为图示方便，以起点固定的四城市 TSP 问题为例。如将起点定为城市 1，并在离散空间图示中省略的话，则所有可能的问题解集在此问题所对应的离散空间内就可表示为 $A \sim F$ 六个点（其中 X、Y、Z 坐标轴分别代表第 1 ~ 3 次选择的城市序号）。其中每个点在各坐标轴上的投影分量就对应于该种城市次序组合所对应的各步分量的城市序号值，而每个点又对应于 TSP 问题的一种解答结果。对应于较多城市数量 N 的 TSP 问题，虽无法用图示方法形象表达，但每个可能的城市组合也必然对应于 N 维空间中的一个点，而该点也必然与该问题的某个解答相对应。

求解 TSP 问题的目标就在于在总数为 N! 的离散点（即问题的可能解的总数）中，以较小的搜索代价寻求最短路径所对应的点（或点集）。对应于起点固定的四城市 TSP 问题，当 $N>4$ 时，如果用离散空间内的蚁群算法求解，在第一步行走时，由于每条路径上都无蚁群留下的信息量，蚁群选择每条路径的概率都相等。对应到图示点中，即其中 $A \sim F$ 每个点被选中的概率都相等。但由于每个点所对应的路径总长不等，根据蚁群算法，较短路径上蚁群所留的信息量就较多。对应到图 7 - 1 中，即与较短路径所对应的点上留存的信息量较多。则蚁群在下一步选择时，就会以更大的概率选取这些点。如此循环往复，蚁群算法就会以越来越大的概率选中与最优解相对应的点，从而求得 TSP 问题的最优解答。而当蚁群数量等于城市数 N 时，每次选择的 N 个点远远无法与解答总数 N!

相比,则此 N 个点对最终解的影响就在于根据相应路径的长短进行 N 点的气味留存之后,其每点所对应的 N 个分量分别对下一步选择作出影响,使较优解答所对应的每步选择分量在下一步以更大概率被选中,从而影响最优路径组合的选择。这时,每一步所选择的点就不一定处于上一步选择的点集之中,上一步所对应的点集是通过各解的优劣和信息量留存对各分量的影响而影响下一步点集的选取的。在离散空间优化问题中,蚁群算法的信息量留存、增减和最优解的选取,都是通过离散的点状分布求解方式而进行的。

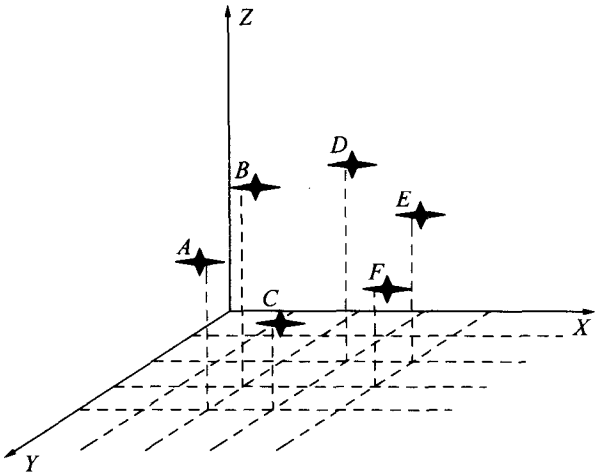


图 7-1 具有固定出发点的四城市 TSP 问题在离散空间的表达示意图

7.2.2 用于连续空间寻优的蚁群算法定义原则

由于在连续空间的寻优问题求解中,解空间是一种区域性的表示方式,而不是以离散的点集方式表示的,所以,连续空间寻优蚁群算法与离散空间寻优蚁群算法之间至少应有以下三个方面的不同:

(1) 每一步求解过程中的蚁群信息量留存方式不应是针对离散的点集或点集分量,而应在对当前蚁群所处点集作出影响的同

时,对这些点的周围区域也有相应的影响。这样,对蚁群信息量的留存方式描述应采用分布函数的形式,其峰值应与当前蚁群所处位置对应的寻优目标函数取值相关。

(2) 蚁群在解空间中的寻优方式不应是在离散解空间点集之间跳变进行,而应是一种微调式的行进方式。

(3) 由于连续空间求解的蚁群信息留存及影响范围是区间性的,而非点状分布,所以在连续空间问题求解中,蚁群判断行进方式所依据的应是总体信息量在与蚁群当前位置所对应的特定区间内的积分累计比较值,而非在各点或点集上的信息量大小。

总体而言,在连续空间内蚁群算法的寻优过程如下:

第一步,使蚁群按一定方式分布于问题所对应的连续空间内(我们推荐均匀分布方式),这样就可求得在蚁群初始分布的离散点处,各单蚁对应于相关问题的初始解分布,同时也得到了对各初始解按最优解标准的评价结果。

第二步,根据具体问题求解的要求,按照相应的标准设计合适的信息量分布函数,使其峰值的大小对应于当前各单蚁所在解空间位置的优劣,使对应于较优解的单蚁的信息量分布函数峰值较高。

第三步,在得到各单蚁相应的信息量分布函数后,将各分布函数的总和对应于按蚁数划分的问题求解子空间进行积分求和,并与总的信息量分布函数在整个问题空间的积分值相比,求得各子空间当前所应有蚁数的比例值和相应于当前蚁群规模的实际蚁数。

第四步,按照一定顺序将当前所考察之蚁所处子空间及临近子空间的应有蚁数与实际蚁数相比较,根据比较结果决定当前考察之蚁的移动方向。

对应于移动后蚁群中各单蚁所对应的解空间位置优劣,就可回到第三步,然后依次进行相应的信息量分布积分、判别和蚁群移动操作。如此循环往复,使整个蚁群按照信息量分布现状所得的启发信息进行合理移动,最终趋于问题的最优解。

7.2.3 用于连续空间寻优的蚁群算法定义

下面,我们对用于连续空间内寻优问题求解的蚁群算法进行定义,然后以连续空间内的函数寻优和系统辨识问题求解为例,举例说明单维和多维连续空间内蚁群算法的应用。

这里,仅以一维空间函数 $y=f(x)$ 的最大值(最小值)寻优为例进行论述,对于多维空间内的函数寻优和系统辨识,只需在此基础上作相应扩展。

所定义的寻优方法如下:

第一步 将蚁群在解空间内按照一定方式作初始分布(推荐均匀分布)。

根据问题定义域的大小,决定合适的蚁群规模,即蚁数 N 的大小。然后将问题的定义域进行 N 等分,并在 N 个子区间的中部放上一只单蚁 $i(i=1\sim N)$。而每只单蚁又带有一个随自己坐标位置变化的移动子区间,自己处于该移动子区间的正中。各移动子区间的长度与问题定义域的 $1/N$ 相等,即将定义域 N 等分后所得的子区间长度相等。当各单蚁处于各子区间的中间位置时,定义此时各子区间内的蚁数为 1。而当各单蚁移动时,根据其所带移动子区间与相邻两子区间的重叠程度变化,定义这两个相邻子区间内的实际蚁数变化。

根据以上定义可知,如果问题的定义域为 $[Start, End]$,则当蚁数为 N 时各子区间长度 D_{RL} 为:$D_{RL}=\dfrac{End-Start}{N}$。由以上描述可知,每只单蚁所带的移动子区间长度 $D_{MRL}=D_{RL}$。而蚁群的初始坐标 x_i 的分布为:$x_i=Start+\left(\dfrac{i}{N}-\dfrac{1}{2}\right)D_{RL}$,其所处子区间 i 的左边界为 $x_{iL}=Start+(i-1)D_{RL}$,右边界为 $x_{iR}=Start+iD_{RL}$。当单蚁移动 Δx 时,由于相邻的子区间与其所带移动区间的重合度变化 Δx,则定义此时相邻两区间内相应于此单蚁移动的实际蚁数 N_{iR} 变化

$\Delta n\left(\Delta n=\frac{\Delta x}{D_{MRL}}=\frac{\Delta x}{D_{RL}}\right)$，即向右移动时，右边子区间内的实际蚁数增加 Δn，而左边子区间内的实际蚁数减少 Δn。向左移动则反之。

第二步 根据蚁群所处解空间位置的优劣，决定当前蚁群的信息量分布。

根据蚁群当前位置 x_i 处的函数值大小 $f(x_i)$，按照寻优问题类别的不同，决定其所留下的相应信息量分布函数的峰值 M_i 的大小，并给出相应的信息量分布函数。例如，如果是特定区间内的函数最小值寻优，可定义相应的信息量分布函数峰值为：$M_i=C-f(x_i)$[式中，C 为根据具体 $f(x_i)$ 的大体范围所设定的常数，满足 $C>f(x_i)$]。这样，对应于较小的函数值，其信息量分布函数的峰值反而大。而如果是函数最大值的寻优，当 $f(x_i)>0$ 时，则可定义 $M_i=C_1f(x_i)$（式中，C_1 为根据具体问题而设定的正常数），而当 $f(x_i)<0$ 时，可定义 $M_i=\frac{C_3}{C_2-f(x_i)}$（$C_2$、$C_3$ 的设定同上）。

对于一维空间内的函数寻优问题，定义单蚁所对应的信息量分布函数为：$T_i(x)=\frac{M_i e^{-k_i(x-x_i)}}{\left[1+e^{-k_i(x-x_i)}\right]^2}$。这样的信息量分布函数呈草帽形，其峰值为 M_i，中心点偏移值为 x_i，而 k_i 为根据实际问题所定义的波形压缩系数。

第三步 根据当前蚁群散布的总信息量分布情况和上一循环中信息量的遗留和挥发情况，决定各子区间内应有的蚁数。

首先，求得当前蚁群散布的总信息量分布函数在各子区间内的积分值 $IN_i(I=1\sim N)$：

$$IN_i=\int_{x_{iL}}^{x_{iR}}\sum_{i=1}^{N}T_i(x)\,dx$$

各子区间的实际总信息量 I_i 应为当前蚁群在该子区间内散布的信息量（IN_i）加上上一次总信息量的遗留部分（ηI_{iLast}，η 为信息量

留存系数),再与所设定的信息量挥发常量 E_V 相减所得的结果:

$$I_i = IN_i + \eta I_{iLast} - E_V$$

然后,求取实际总信息量在整个问题区间的总和值 I_Σ:

$$I_\Sigma = \sum_{i=1}^{N} I_i$$

这样,根据各子区间实际总信息量 I_i 占 I_Σ 之比例,可求得当前蚁群分布条件下决定的各子区间应有蚁数 N_{iM}:

$$N_{iM} = \frac{I_i}{I_\Sigma} \cdot N$$

在实际的编程运算中,由于我们所取的信息量分布函数 $T_i(x)$ 为可积函数,其原函数 $\prod_i(x)$ 为 $\prod_i(x) = \int T_i(x)\,dx = \frac{M_i}{1+e^{-k_i(x-x_i)}}$,其中 $I=1\sim N$。这样,积分值 IN_i 的求取可直接利用各原函数 $\prod_i(x)$ 在被积区间边界点上的取值相减所得结果进行求和即可,即:$IN_i = \sum_{i=1}^{N} [\prod_i(x_{iR}) - \prod_i(x_{iL})]$。

第四步 根据各子区间内应有的蚁群分布状况和当前蚁群分布状况之间的差别,决定蚁群的移动方向,并加以移动。

首先,根据已求得的各子区间内的应有蚁数 N_{iM},以所考察之蚁当前所处区间为界进行求和操作,求出被考察之蚁所处区间 i 之左的应有蚁数之和 N_{iML},和所处区间 i 之右的应有蚁数之和 N_{iMR},作为被考察之蚁移动方向判定的依据条件:

$$N_{iML} = \sum_{j=1}^{i-1} N_{jM}, N_{iMR} = \sum_{j=i+1}^{N} N_{jM}$$

同样,还需根据已知的各子区间内实际蚁数 N_{iR},以所考察之蚁当前所处区间为界进行求和操作,求出被考察之蚁所处区间 i 之左的实际蚁数之和 N_{iRL},和所处区间 i 之右的实际蚁数之和 N_{iRR}:

$$N_{iRL} = \sum_{j=1}^{i-1} N_{jR}, N_{iRR} = \sum_{j=i+1}^{N} N_{jR}$$

然后，根据被考察之蚁所处区域及其左右的实际蚁数和应有蚁数之间的差别，决定该蚁的运动方向，并作 Δx 的坐标变化。其运动规则如下表所示：

表 7-1 蚁群运动规则

规则	$N_{iRL} < > N_{iML}$	$N_{iR} < > N_{iM}$	$N_{iRR} < > N_{iMR}$	被考察之蚁的坐标变化值
1	<	=	>	$-\Delta x$
2	>	=	<	$+\Delta x$
3	=	>	<	$+\Delta x$
4	>	>	<	$+\Delta x$
5	<	>	=	$-\Delta x$
6	<	>	>	$-\Delta x$
7	<	>	<	依次 $+/-\Delta x$

其他情况下被考察之蚁均不动。

其中，为执行简便起见，每当出现规则 7 的情况时，不妨规定，将符合条件之蚁依次进行坐标的左移和右移操作，并设置相应的标志位，作为下一次移动方向的判定依据。

在蚁群作完一次整体移动之后，又可回到第二步，进行相应的信息量分布、考察和蚁群移动操作，如此循环往复，直到最优解的产生。

如果严格按照以上模式操作，可以发现，当整个蚁群都运算收敛至同一个子区间时，如果仅以此子区间为基准进行判别操作，会出现整个蚁群停滞不前的情况。举例如下：

当整个蚁群都处于子区间 i，而 $N_{iRL} > N_{iML}, N_{iR} > N_{iM}, N_{iRR} > N_{iMR}$ 时，按照前面的移动规则，整个蚁群就永远处于停滞状态。这是与寻优的收敛要求不符的。

为此，我们考察蚁群的移动方向，不仅仅以各单蚁坐标所处区

间为准进行考察，而且同时在与各单蚁所带移动区间相交的两个子区间内进行考察，按照考察的结果决定被考察之蚁的移动方向。这样，当出现上述情况时，虽然在区间 i 的考察都判令蚁群不动，但在蚁群相邻区间的考察必然判令蚁群作相应的移动，从而进一步促进寻优进程。

7.2.4 函数寻优实例研究

应用(1) 多极值函数的最小值寻优问题求解

被寻优的函数为：

$$y = f_1(x) = 5x^6 - 36x^5 + 82.5x^4 - 60x^3 + 36$$

这是一个多极值函数，已知其局部最优点位于 $x=1$、$x=2$ 和 $x=3$ 处，其中，在 $x=2$ 时函数取局部最大值，在 $x=1$ 和 $x=3$ 处函数取局部最小值，而在 $x=3$ 处为全局最小值。函数的寻优区间为 $[0,3.5]$（如图 7-2 所示）。

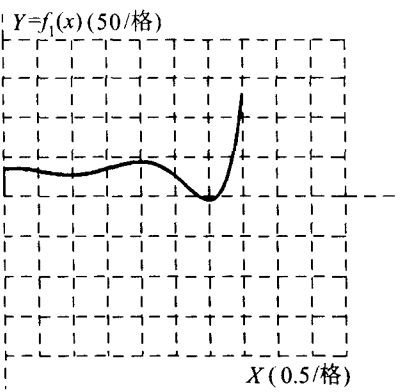


图 7-2 用于蚁群算法寻优的多极值函数

为表征算法的收敛特征，定义该例中蚁群的总体求解误差为：

$$E_\Sigma = \sum_{i=1}^{N} |x_i - 3|$$

这里，所需选择的蚁群算法参数为蚁群规模（即蚁数 N），

信息量分布函数的压缩系数 k_i，信息留存系数 η，蚁群寻优步长 Δx，信息量挥发常数 E_V 等。各信息量分布函数的峰值 M_i 取为 $[C - f(x_i)]$ 形式。

总体而言，整个蚁群都能从初始的均匀分布状态按照所给算法以一定精度趋于最优解的附近。当求解空间随寻优过程的进行逐步缩小时，经五次寻优过程之后，其最终的总体求解误差 $E_\Sigma = 0.037$，即每只单蚁的平均求解误差为 0.14%（在五次寻优过程中，$N = 9$，循环次数均取 500 次，η 均取 0.01，E_V 均取 50，M_i 表达式中的常数 $C = 150$）。

典型的五次寻优过程的蚁群算法参数设定列表如下：

表 7-2 典型的五次寻优过程中蚁群算法的参数设定

次数	区间设定	压缩系数 k_i	寻优步长 Δx	总体求解误差 E_Σ	经所设循环次数后蚁群所处的解答空间分布
1	[0,3.5]	6	0.008	1.29	[2.6,3.2]
2	[2.4,3.4]	1.8	0.008	0.884	[2.8,3.1]
3	[2.7,3.2]	2.8	0.004	0.297	[2.95,3.05]
4	[2.9,3.1]	6.9	0.004	0.096	[2.97,3.02]
5	[2.95,3.05]	14.05	0.003	0.037	[2.99,3.01]

五次寻优过程中，代表性的总体求解误差变化（第一次寻优）如图 7-3 所示。

应用(2) 非线性函数的最大值寻优问题求解

被寻优的非线性函数为（如图 7-4 所示）：

$$y = f_2(x) = 3x^2 e^{-x}$$

这是一个非线性的单极值函数，在 $x = 2$ 处取得全局最大值 1.6240，寻优区间为[0,3]。同样，定义蚁群的总体求解误差为：

$$E_\Sigma = \sum_{i=1}^{N} |x_i - 2|$$

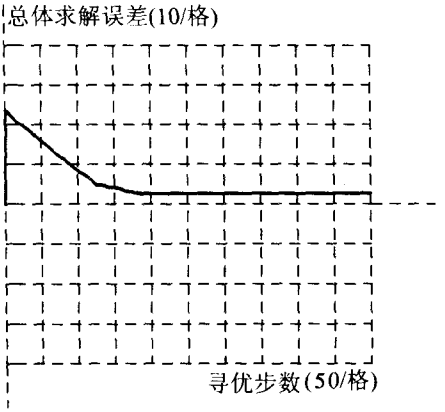


图 7-3 多极值函数第一次寻优过程的误差变化动态

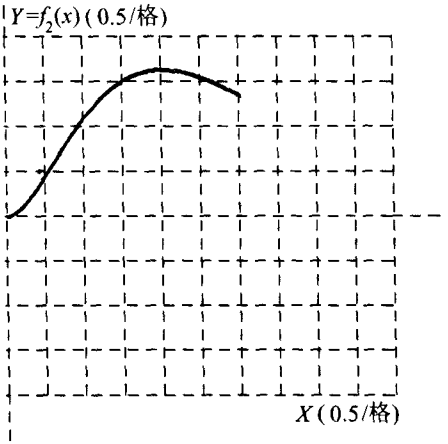


图 7-4 用于蚁群算法寻优的非线性函数

这里,令各信息量分布函数的峰值形式为 $Cf(x_i)$。在选取蚁数 $N=9$,信息留存系数 $\eta=0.01$,寻优步长 $\Delta x=0.0002$,信息量挥发常数 $E_V=65$,信息量分布函数峰值表达式中的常数项 $C=200$ 之后,整个蚁群算法只需对信息量分布函数的压缩系数

k_i 进行调整，就能使蚁群进入良好的收敛状态（循环次数统一为8000次）。当求解空间随寻优过程的进行逐步缩小时，经四次典型求解过程之后，其最终的总体求解误差 $E_\Sigma = 0.055$，即每只单蚁的平均求解误差为0.31%。

典型的四次寻优过程如下：

表7-3 典型的四次寻优过程中蚁群算法的参数设定

次数	寻优区间设定	压缩系数 k_i	总体求解误差 E_Σ	经所设寻优次数后蚁群所处的解答空间分布
1	[0,3]	20	1.347	[1.75,2.14]
2	[1.5,2.4]	180	0.450	[1.91,1.98]
3	[1.8,2.1]	580	0.150	[1.97,1.99]
4	[1.9,2.05]	580	0.055	[1.99,2.01]

四次寻优过程中，典型的总体求解误差变化（第一次寻优过程）如图7-5所示。

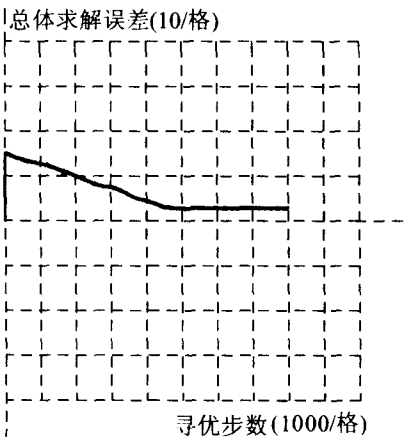


图7-5 非线性函数第一次寻优过程的误差变化动态

7.3 用于多维连续空间内系统参数辨识的蚁群算法

7.3.1 基本模式

为叙述和图示简便起见，我们以线性系统 $\dot{X} = A_p X + B_p U$ 的参数辨识为例，说明蚁群算法在系统辨识领域的应用。在此辨识问题中，蚁群算法寻优的空间维数为矩阵 A_p 和 B_p 的元素个数之和，即所需辨识的参数个数之和。在以下的论述中，我们假设矩阵 A_p 和 B_p 均为单元素，即蚁群个体是在 A 轴和 B 轴所构成的二维参数空间内进行参数辨识操作。

该蚁群辨识算法实际上将前一部分所述的用于单变量函数寻优的蚁群算法作相应的维数扩充和修正即可（总体框图见图 7-6，其中被辨识系统参数为 A_p 和 B_p，而参照系统参数 A_{Si} 和 B_{Si} 的变化受蚁群在解空间内的寻优移动过程制约）。

其总体求解步骤如下：

第一步　将蚁群在解空间内按照一定方式作初始分布（推荐均匀分布）。

这里，也需根据问题定义域的大小，即被辨识系统参数的可能范围的大小，决定合适的蚁群规模。我们推荐的蚁群规模为 N^2 只，即以每 N 只单蚁为一组，在 A 轴方向上作均匀分布，而同一组内的 N 只单蚁又按照 B 轴方向在问题空间内作均匀分布。举例来说，如果被辨识系统问题的定义域为 A：[$StartA$, $EndA$]，B：[$StartB$, $EndB$]（矩形形状），同时，蚁群分布又按照先变化 B_{Si}，后变化 A_{Si} 的均匀分布方式，则蚁群的初始坐标分布为：

$$A_{Si} = StartA + \left[Int\left(\frac{i-1}{N}\right) + \frac{1}{2} \right] D_{AL}$$

$$B_{Si} = StartB + \left[i - Int\left(\frac{i-1}{N}\right) N - \frac{1}{2} \right] D_{BL}$$

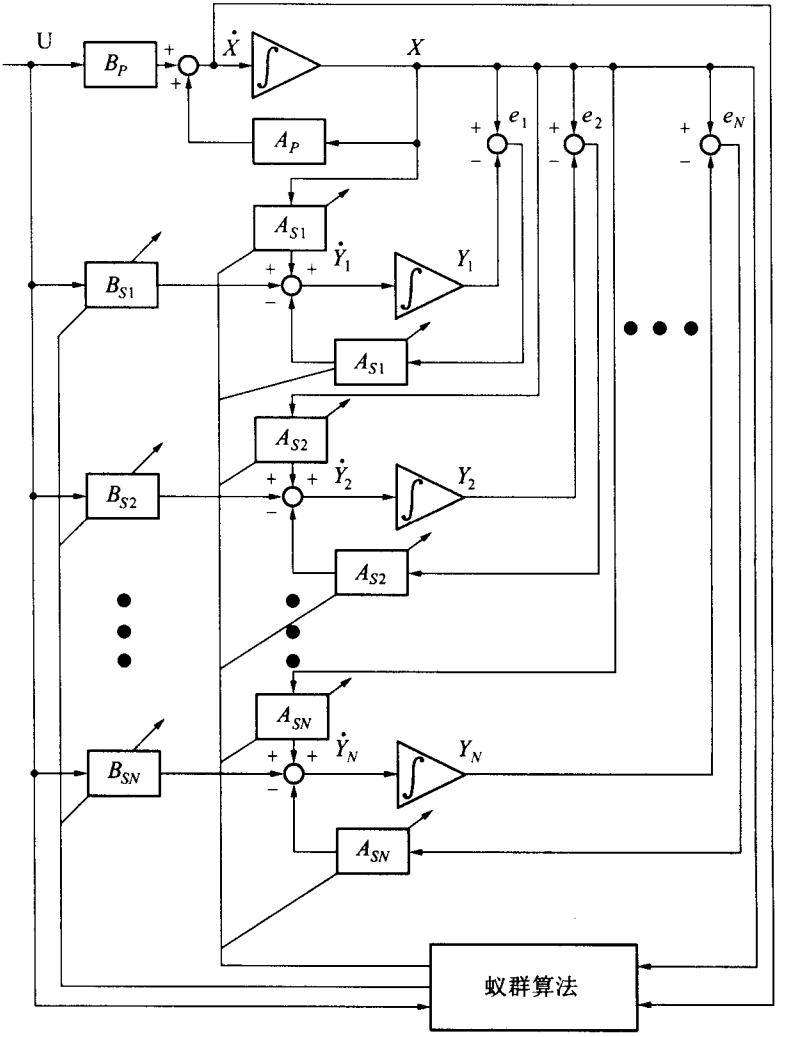


图7-6 蚁群算法用于线性系统参数辨识的总体框图

[Int( )为取整函数]

式中,D_{AL}和D_{BL}为将问题空间在A轴和B轴上的映射进行N等分后所得的单维区间长度。即:

$$D_{AL} = \frac{EndA - StartA}{N}, D_{BL} = \frac{EndB - StartB}{N}$$

这些区间在A轴和B轴上映射的左边界A_{iL}、B_{iL},右边界A_{iR}、B_{iR}取值分别为:

$$A_{iL} = A_{Si} - D_{AL}/2, B_{iL} = B_{Si} - D_{BL}/2$$

$$A_{iR} = A_{Si} + D_{AL}/2, B_{iR} = B_{Si} + D_{BL}/2$$

这样,每只单蚁就处于将参数空间的A轴分量和B轴分量在参数变化范围内均进行N等分后所得到的N^2个矩形子空间的中心。同样,每只单蚁也均带有一个随自己坐标位置变化的移动矩形子空间,而自己处于该移动的矩形子空间的中心。在这里,移动矩形子空间的长度和宽度分别为D_{AL}和D_{BL},定义此时各矩形子空间内的蚁数为1。当各单蚁移动时,根据其所带移动矩形子空间与相邻子空间的重叠程度变化,定义相邻子空间内的实际蚁数变化。例如,如果单蚁i从(A_{Si}, B_{Si})移动至$(A_{Si} + \Delta A, B_{Si} + \Delta B)$,则移动后,与此单蚁所带移动子空间相交的$A$轴和$B$轴子空间的蚁数就相应变化$\Delta n_A = \frac{\Delta A}{D_{AL}}$和$\Delta n_B = \frac{\Delta B}{D_{BL}}$。

第二步 根据蚁群所处解空间位置的优劣,决定当前蚁群的信息量分布。

这里,蚁群信息量分布函数的定义要与各单蚁当前所处解空间位置(A_{Si}, B_{Si})针对参数辨识正确性的优劣相关。在二维空间内,我们定义各单蚁所对应的信息量分布函数为:

$$T_i(A_S, B_S) = \frac{M_i e^{-k_i\sqrt{(A_S - A_{Si})^2 + (B_S - B_{Si})^2}}}{\left[1 + e^{-k_i\sqrt{(A_S - A_{Si})^2 + (B_S - B_{Si})^2}}\right]^2}$$

此函数实际上是将一维空间内函数寻优所用的信息量分布函

数 T_i 扩展至二维空间(A,B)，并以(A_{Si},B_{Si})为中心作相应旋转操作后所得，其形状如图7-7所示。

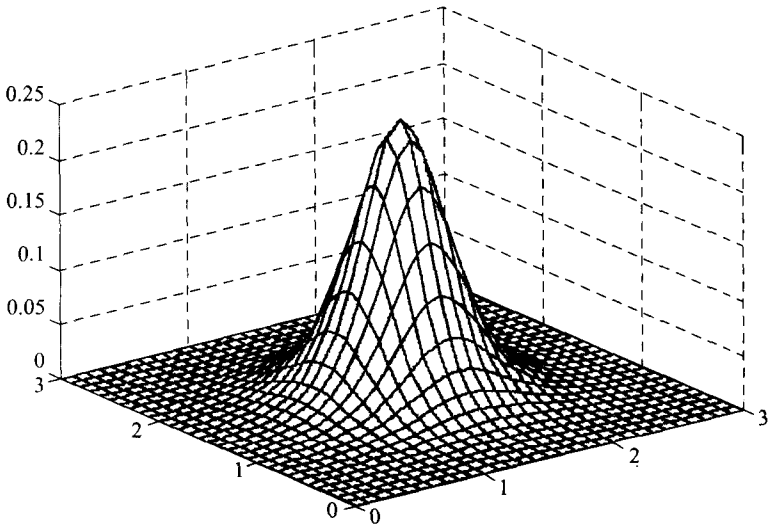


图7-7 以当前蚂蚁位置为中心的信息量分布函数示例
（当前蚂蚁位置为[1.5,1.5,0]）

同样，这里的 k_i 为根据实际问题所定义的压缩系数。其峰值 M_i 的定义要与各单蚁当前所处的解空间位置(A_{Si},B_{Si})的优劣相关，即：

$$M_i = [C_4 - (\dot{x} - A_{Si}x - B_{Si}U)^2]/C_5$$

式中，$\dot{x}$，x，U 为被辨识系统当前检测的状态量变化率、状态量，以及外加输入值。当(A_{Si},B_{Si})接近于(A_p,B_p)时，以上所定义的信息量分布函数峰值 M_i 显然取值较高。

第三步 根据当前蚁群散布的总信息量分布情况和上一循环过程中信息量的遗留和挥发情况，决定各子区间内应有的蚁数分布。

先针对 A 轴进行以上操作。

首先,求得当前蚁群散布的总信息量分布函数在各 A 轴子区间内的积分值 $IN_{iA}(i=1\sim N)$(此时,B 轴的积分边界取[$StartB$, $EndB$]):

$$IN_{iA} = \int_{StartB}^{EndB}\int_{A_{iL}}^{AR}\sum_{i=1}^{N^2} T_i(A,B)\,dA\,dB$$

各 A 轴子区间的实际总信息量 I_{iA} 应为当前蚁群在该 A 轴子区间内散布的信息量(IN_{iA}),加上上一次总信息量的遗留部分($\eta \cdot I_{iALast}$,η 为信息量留存系数),再与所设定的信息量挥发常量 E_V 相减所得的结果:

$$I_{iA} = IN_{iA} + \eta I_{iALast} - E_V$$

然后,求取实际总信息量在整个 A 轴问题区间的总积分值 $I_{\Sigma A}$:

$$I_{\Sigma A} = \sum_{i=1}^{N^2} I_{iA}$$

根据各 A 轴子区间实际总信息量 I_{iA} 占总积分值 $I_{\Sigma A}$ 的比例,可求得当前蚁群分布条件下决定的各 A 轴子区间内应有蚁数 N_{iMA}:

$$N_{iMA} = \frac{I_{iA}}{I_{\Sigma A}}N^2,(i = 1 \sim N^2)$$

同样,在实际的编程运算中,由于我们所取的信息量分布函数 $T_i(A_S,B_S)$ 为可积函数,其原函数 $\prod_i(A_S,B_S)$ 为:

$$\prod_i(A_S,B_S) = \iint T_i(A_S,B_S)\,dA_S\,dB_S$$

$$= M_i\left[\frac{-k_i\sqrt{(A_S-A_{S_i})^2+(B_S-B_{S_i})^2}}{1+e^{-k_i\sqrt{(A_S-A_{S_i})^2+(B_S-B_{S_i})^2}}} + \ln\left(1+e^{k_i\sqrt{(A_S-A_{S_i})^2+(B_S-B_{S_i})^2}}\right)\right]\operatorname{arctg}\frac{B_S-B_{S_i}}{A_S-A_{S_i}}$$

这样,对积分值 IN_{iA} 的求取可得到一定程度的简化。在我们的编程运算中,为进一步加快运算速度,积分值的求取采用在相应积分区间内均匀地取若干点,然后以这些点处函数值的平均与相

应的区间面积相乘的方法。当被积函数曲面连续平滑时，用这样的方法的拟合效果较好，并且节省了运算代价。

针对 B 轴各子空间的运算操作与 A 轴类似（其中，在针对 B 轴各子区间的积分值求取时，A 轴分量的积分边界取[$StartA$, $EndA$]）。

第四步 根据各子区间内应有的蚁群分布状况和当前蚁群分布状况之间的差别，决定蚁群的移动方向，并加以移动。

同样，先针对 A 轴进行以上操作。

首先，根据已求得的 A 轴各子区间内的应有蚁数 N_{iMA}，以所考察之蚁当前所处的 A 轴区间为界进行求和操作，求出被考察之蚁所处 A 轴区间 i 之左的应有蚁数之和 N_{iMLA}，和所处 A 轴区间 i 之右的应有蚁数之和 N_{iMRA}，作为被考察之蚁移动方向判定的依据条件：

$$N_{iMLA} = \sum_{j=1}^{i-1} N_{jMA}, N_{iMRA} = \sum_{j=i+1}^{N} N_{jMA}$$

同样，还需根据已知的 A 轴各子区间内的实际蚁数 N_{iRA}，以所考察之蚁当前所处 A 轴区间为界进行求和操作，求出被考察之蚁所处 A 轴区间 i 之左的实际蚁数之和 N_{iRLA}，和所处 A 轴区间 i 之右的实际蚁数之和 N_{iRRA}：

$$N_{iRLA} = \sum_{j=1}^{i-1} N_{jRA}, N_{iRRA} = \sum_{j=i+1}^{N} N_{jRA}$$

然后，根据被考察之蚁所处 A 轴子区间及其左右的实际蚁数和应有蚁数之间的差别，决定该蚁的运动方向，并将该蚁的 A 轴坐标值变化 ΔA。其运动规则如下表所示：

其他情况下被考察之蚁的 A 轴坐标均不作变动。

同样，为执行简便起见，每当出现规则7的情况时，我们规定，将符合条件之蚁依次进行 A 轴坐标的左移和右移操作，并设置相应的标志位，作为下一次移动方向的判定依据。

表 7-4 蚁群运动规则

规则	$N_{iRLA} < > N_{iMLA}$	$N_{iRA} < > N_{iMA}$	$N_{iRRA} < > N_{iMRA}$	被考察之蚁的 A 轴坐标变化
1	<	=	>	$-\Delta A$
2	>	=	<	$+\Delta A$
3	=	>	<	$+\Delta A$
4	>	>	<	$+\Delta A$
5	<	>	=	$-\Delta A$
6	<	>	>	$-\Delta A$
7	<	>	<	依次 $+/-\Delta A$

类似地，为避免整个蚁群处于同一子区间时可能出现的停滞状态，我们规定，决定蚁群在 A 轴方向的移动，不仅以各单蚁 A 轴坐标所处子区间为准进行考察，而且同时在与各单蚁所带移动子区间相交的两个 A 轴子区间内进行考察，按照考察结果决定被考察之蚁的移动方向。

蚁群 B 轴分量的变化寻优过程与 A 轴分量类似。

在蚁群作完一次整体移动之后，又可回到第二步，进行相应的信息量分布、考察和蚁群移动操作，如此循环往复，直到最优解的产生。

7.3.2 实例研究

令被辨识系统为 $\dot{X} = A_p X + B_p U$，A_p 和 B_p 为单参数，其中 $A_p = B_p = 2$，U 取阶跃式输入（幅值为 8）。假设系统的状态变量 x 及其变化率 $\dot{x}$，以及外界输入 U 的实际值以 $\Delta t = 0.00001\text{s}$ 的时间间隔被准确采样，作为蚁群辨识系统的输入。则按照第三部分所给出的系统辨识蚁群算法，在信息留存系数 $\eta = 0.01$，信息量分布函数压缩系数 $k_i = 1$，寻优步长 $\mathrm{d}A = \mathrm{d}B = 0.01$，信息量峰值表达式

中 $C_4 = 1000, C_5 = 5$，信息量挥发系数 $E_V = 228$，初始蚁群以 $N^2 = 4^2 = 16$ 的蚁数均匀分布于 $A_S:[0,5], B_S:[0,5]$ 的矩形空间内时，经9250步辨识寻优后，整个蚁群的辨识总误差（$E_\Sigma = E_{\Sigma A} + E_{\Sigma B} = \sum_{i=1}^{16} | A_{Si} - A_p | + \sum_{i=1}^{16} | B_{Si} - B_p |$）为14.43。第一次寻优结果是，整个蚁群在辨识寻优空间的取值为 $A_{Si}:[1.265, 3.265]$，$B_{Si}:[1.425, 3.205]$ 之间。第一次辨识寻优过程的总误差变化动态如图7-8所示。以相同的蚁群辨识系统参数设计，当辨识寻优区间根据第一次寻优结果缩小为 A, B 均为[1,3.5]时，经9250步辨识寻优后，蚁群的总辨识误差为 $E_\Sigma = 6.37$。第二次辨识寻优结果区间分布为 $A_{Si}:[1.76, 2.30], B_{Si}:[1.99, 2.68]$，其总辨识误差的变化动态如图7-9所示。第三次辨识寻优时，当寻优区间缩小为 A, B 均为[1.5,2.8]，在 $E_V = 228, k_i = 1.2$ 时，蚁群经9200步辨识寻优后，收敛至总辨识误差 $E_\Sigma = 2.75$，其变化动态如图7-10所示，寻优结果区间分布为 $A_{Si}:[1.91, 2.26], B_{Si}:[1.84, 2.20]$。当第四次辨识寻优时，寻优区间缩小为 A, B 均为[1.7,2.4]，$k_i = 1.4$，

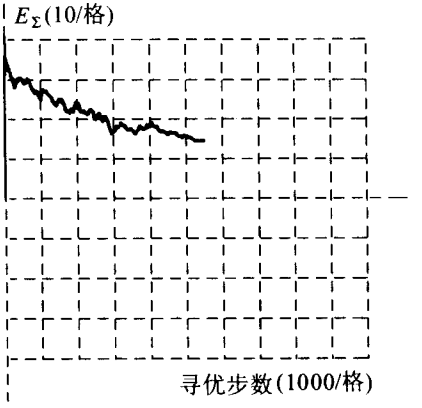


图7-8 第一次寻优过程的误差变化动态

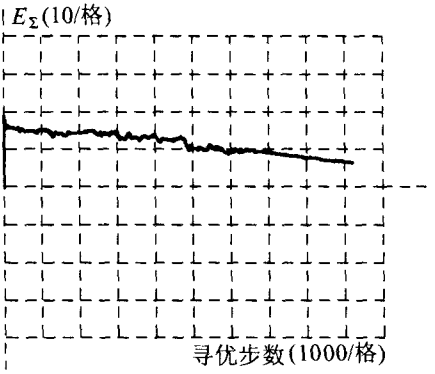


图 7-9 第二次寻优过程的误差变化动态

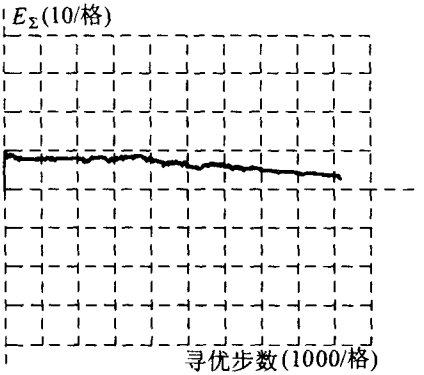


图 7-10 第三次寻优过程的误差变化动态

$E_V = 128, C_4 = 1500$ 时，经 11000 步辨识寻优后，E_Σ 收敛至 1.445，而寻优结果区间为 A_{Si}：[1.948,2.218]，B_{Si}：[1.918,2.098]，总的辨识误差变化动态如图 7-11 所示。而当第五次辨识寻优区间压缩至 A, B 均为[1.8,2.2]时，由于寻优区间大大缩小，则对蚁群系统各参数的调整幅度也大大增加。当 $U = 48$ 时，以

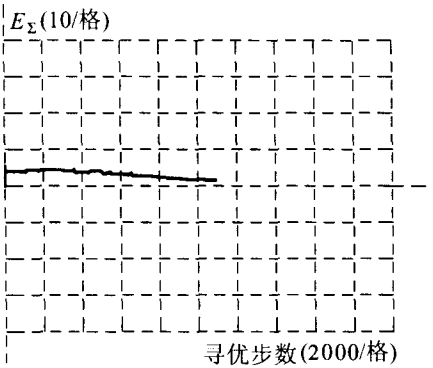


图7-11 第四次寻优过程的误差变化动态

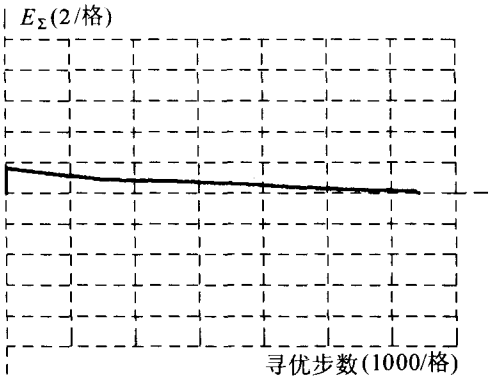


图7-12 第五次寻优过程的误差变化动态

$dt = 0.00005$ 的系统准确检测信息输入个数为 $N^2 = 3^2 = 9$ 的蚁群系统，当 $\eta = 0.51$，$k_i = 12$，$E_V = 78$，$dA = dB = 0.00002$，$C_4 = 8000$ 时，经6400步辨识寻优后，整个蚁群的辨识寻优结果分布区间为 A_{Si}：[1.9912，2.0037]，B_{Si}：[1.9824，2.0000]，总辨识误差 E_Σ 收敛至0.118627，其变化动态如图7-12所示。此时，整个蚁群

的平均参数辨识精度已达0.33%。

7.4 蚁群算法的总体特征及相关参数选取原则

上面所举的三个实例仿真结果,以及仿真实验中的蚁群算法参数调整过程,使我们对蚁群算法在连续空间内进行问题求解的总体寻优特征有了深入的了解。在连续空间内的优化问题求解中,各单蚁智能体通过散布与所在解空间位置优劣程度相关的信息量分布函数来对蚁群的总体运动方向作出影响。而蚁群的总体运动方向是在对特定区域内整个蚁群的信息量分布状态进行考察之后决定的。蚁群运动的总体效果是在连续的解空间内逐步收敛至最优解所在的邻近区域。各单蚁的信息量分布函数对整个解空间所处区域均有影响,只是影响程度随离各单蚁所在解空间位置距离的增加而递减。这样,为避免蚁群初始分布不均对整个问题求解效果的影响,避免蚁群最终趋于问题的局部最优解,我们推荐在问题求解之初,使蚁群在解空间内作均匀分布,而判定蚁群运动方向则依据以各单蚁所在子区域为界的总体信息量分布状况。

在蚁群算法的有关各参数选择中,我们认为,信息量挥发常数E_V的设定是至关重要的。该参数通过在蚁群当前散布的总信息量中减去一个常数,可以提高具有较高数值的子区域信息量累计值占总的问题空间信息量比例,从而影响下一步蚁群运动方向的选择。该参数选择过低,则对寻优过程的促进效果不明显;而如果选择过高,则可能超出某些区域内信息量的积分总和,对整个蚁群的运动判别造成不利影响,产生早熟性的收敛结果,甚至使整个蚁群收敛过程过早停滞。由于我们所给出的算法中,E_V对各子区间的信息量分布有直接影响,所以,子区间的划分,即蚁群的规模N(或N^2)、蚁群信息量分布函数的峰值及压缩系数的选择都与之相关。一般而言,蚁群规模越小,各信息量分布函数对应子区间范围越大,各压缩系数设置越小,即信息量分布函数变化越平缓,则E_V需

设得越大，以使信息量在各子区间范围内的分布情况变化足以影响蚁群的运动状态选择。

而信息留存系数 η 的大小对蚁群算法的寻优效果影响并不明显。这是因为，不论 η 取值如何，它都是按照相应的设定比例将前一次的信息量分布影响下一次的信息量分布状况，而并未对各子区域内的相对信息留存比例作出优化性的变更。实际的仿真结果也证明了这一点，所以，在大部分的仿真实验中，我们对 η 的取值都是固定的。

另外，蚁群的寻优步长也要根据问题域的大小、蚁群规模的大小而定。一般而言，随着寻优过程的进行和寻优空间的缩小，蚁群的寻优步长也应相应缩小。只有这样，才能进一步提高寻优精度，避免振荡的发生。而压缩系数 k_i 的选择也与寻优空间的大小关系密切，不同的是，它是应随着寻优过程的进行和寻优空间的缩小而增大的。只有这样，才能使各缩小的寻优子空间之间的信息量分布产生足够的差别，从而在蚁群寻优步长减小的条件下也能产生持续的运动导向，进一步促进寻优进程。

对于系统的参数辨识问题，在辨识时段内被辨识系统信息的变化动态也对辨识结果有一定的影响。一般而言，外加的输入幅值越大，输入辨识系统的动态信息越丰富，则越利于蚁群辨识结果趋于正确值。但对此问题的研究已超出了蚁群算法本身的研究范围。

从本质而言，蚁群算法是应以分布式的协同优化计算方式为特征的。所以，在串行计算机上对蚁群算法的模拟代价并不能真正体现蚁群算法的本质特征，因而，进一步的研究工作应针对该算法的并行机实现和协同运算机理研究问题。

第8章　总结

8.1　我们对蚁群算法的认识

从前面所述实例及相关结果，以及蚁群算法各参数的调整过程，大家基本上可以对蚁群算法在离散和连续空间优化中的大致特性有了深刻的认识。蚁群算法之所以能引起相关领域研究者的注意，是因为该种求解模式能将问题求解的快速性、全局优化特征以及有限时间内答案的合理性结合起来。其中，寻优的快速性是通过正反馈式的信息传递和积累来保证的，而其分布式计算的特征又避免了算法的早熟性收敛，同时，具有贪婪启发式搜索特征的蚁群系统又能在搜索过程的早期就找到可以接受的问题解答。而在本书所述的连续空间优化过程中，每一只蚂蚁智能体经过拓展，可以通过对应于与最优解的相符程度的痕迹分布来影响这个蚁群的运动，并且在观察了相关区域的总体痕迹分布之后，合理决定蚁群的运动。蚁群运动的总的效果是以一定精度收敛到连续空间中最优解的附近区域。每一只蚂蚁的痕迹分布函数能影响整个的解空间，当与当前蚂蚁位置的距离增加时，其影响程度就减小。痕迹挥发效果、痕迹分布的峰值以及收缩系数的选择都与优化结果相关，蚁群的优化步骤以及收缩因子的选择也与优化效果密切相关。

蚁群算法从本质上讲是一种模拟进化算法，它的产生与进化算法的发展息息相关。如果与群体搜索策略结合使用，并保证群体中个体之间的信息交换，则蚁群算法可体现进化计算的优越性：首先，进化算法在搜索过程中不容易陷入局部最优，即使在所定义的适应函数是不连续的、非规则的或有噪声的情况下，它们也能以很大的概率找到全局最优解；其次，由于它们固有的并行性，进化

算法非常适合于巨量并行机;再者,进化算法采用自然进化机制来表现复杂的现象,能够快速可靠地解决非常困难的问题。这样,由于它们容易介入到已有模型中,具有可扩展性,以及具有易与别的技术相融合等有利因素,进化算法已经在各相关领域得到了越来越广泛的应用。

本书中,通过对蚁群算法研究历史的回顾和算法综述,并且通过从各典型研究领域及工程领域的应用综述,使我们对蚁群算法在其中的研究和应用成果有了一定的了解。在算法的具体描述和典型优化及应用问题的求解模式设计中,在算法的拓展应用领域,作者讨论了该算法的详细应用和拓展模式。如果本书的出版能够引起相关领域研究人员的关注,并且读者能够从中指出我们研究中存在的问题,这将是作者最大的快乐。

8.2 群体智能——未来的发展方向

群体智能的概念是从包括蚁群算法相关领域的研究中提出的。目前,关于群体智能的研究还处于萌芽阶段,还存在很多问题。首先,它们都是启发式或概率性的算法,从数学上对于它们的正确性与可靠性进行证明还是比较困难的,所做的相关工作也比较少;其次,这些算法都是专用的算法,一个算法一般只能解决某一类问题,各种算法之间的相似性很差;并且,系统的高层次的行为是需要通过低层次的昆虫(可看成智能体)之间的简单行为交互而涌现产生的。单个个体控制的简单性并不意味着整个系统设计的简单性,因此我们必须能够将高层次的复杂行为(也就是系统所要执行的功能,例如旅行商问题、数据聚类、货物搬运等)映射到低层次简单个体的简单行为(例如信息素的遗留、物体的拾起与放下)上面,而这两者之间是存在较大差别的。并且,在系统设计时也要保证多个个体简单行为的交互能够涌现出我们所希望看到的高层次复杂行为,这可以说是群体智能中一个极为困难的

问题。

在目前对何为智能,何为智能自动化的相关讨论尚无定论的时候,我们认为,进行智能自动化及智能计算领域的理论及应用研究,至少应持有以下观点:

(1) 智能具有多样性。智能的多样性,源于物质的多样性,这是世界存在和发展的首要的表现方式。只有在承认存在差别的前提下,才能去讨论各类智能的本体,才能总结出智能最根本的特征和存在方式。

(2) 智能具有发展性,并且是不断发展的。智能的发展特性源于世界的运动特征,它本身就是一个不断演化的概念。如果要智能向着我们所要求的方向发展,首先必须承认这一点,然后才能规划我们所做的工作。

(3) 智能具有局限性。这是在前两个观点的基础上的必然推论。世界上不存在十全十美的智能实现,任何智能控制模式,都有其特定的适用范围,是不能被无原则滥用的。

(4) 智能具有层次性。这是所需求解的问题本身所必然存在的层次性所决定的。

在以上四个基本观点指导下,我们进行了一定的具体研究工作:

在"基于生物智能模拟的智能控制"的相关领域研究中,我们对具有群体智能特征的智能实现模式给予了充分的关注。在此类智能实现模式中,模拟生物系统某类局部智能运行模式的智能个体,其能力相对于所需求解的问题而言,是完全不具有全局性的求解功能的。对于每个智能个体,在其局部动态特征或运动决策功能中所具有的智能性,相对于智能体群所表现的智能特性而言,是完全具有层次性的等级差别的。每个智能个体在其解空间中的单独运动中所表现的智能特性,如果离开了其他智能体的协同工作和具全局性指导特征的信息传递,是无法满足问题求解的需要的。

因此,在此类基于智能体群的智能实现模式下,对智能个体特性的合理定义,智能体间信息交互的模式设计和指导寻优方向的关键指导函数或指导参量设计,是至关重要且互相联系的三个方面,缺一不可。在以智能个体运动规律设计为主体的群体智能实现模式——蚁群算法的研究中,我们正是运用了以上的指导思想。在这种具有启发式群体智能寻优特征的智能算法设计中,其单蚁的运动规则设计及参数选择对寻优结果起着决定性的作用。在具有一定数量智能个体的群体信息量交互环境下,整个蚁群表现出了单蚁所无法完成的寻优动态,并从 TSP 问题求解等典型的离散空间问题求解拓展至连续空间中的非线性、多极值函数寻优以及线性系统的参数辨识问题求解之中,显示了该类算法有吸引力的寻优特征。可以说,本书所总结的全部相关研究结果,均体现了大规模并行演化计算的智能涌现特征,同时,群体智能、演化计算等相关领域也正是当前国际上智能学科的前沿研究领域。如何将其引入智能自动化学科,并作出进一步的创新,是一个具有挑战性的研究方向。

参考文献

陈昌富,谢学斌. 2002. 露天采矿边坡临界滑动面搜索蚁群算法研究. 湘潭矿业学院学报, 17(1): 62 - 64

陈峻,秦玲. 2003. 具有感觉和知觉特征的蚁群算法. 系统仿真学报, 15(10): 1418 - 1425

陈峻,沈洁. 2003. 基于分布均匀度的自适应蚁群算法. 软件学报, 14(8): 1379 - 1387

丁亚平,苏庆德,吴庆生. 2002. 化学蚁群算法与多组分导数荧光光谱解析. 高等学校化学学报, 23(9): 1695 - 1697

冯祖洪,徐宗本. 2002. 用混合型蚂蚁群算法求解 TSP 问题. 工程数学学报, 19(4): 35 - 39

高坚. 2003. 基于自适应蚁群算法的多受限网络 QoS 路由优化. 计算机工程,29(19): 40 - 41

郜庆路,罗欣. 2003. 基于蚂蚁算法的混流车间动态调度研究. 计算机集成制造系统——CIMS, 9(6): 456 - 459

郝晋,石立宝. 2002. 求解复杂 TSP 问题的随机扰动蚁群算法. 系统工程理论与实践, 22(9): 88 - 91

郝晋,石立宝,周家启. 2002. 一种求解最优机组组合问题的随机扰动蚁群优化算法. 电力系统自动化,26(23): 1 - 6

洪炳熔,金飞虎. 2003. 基于蚁群算法的多层前馈神经网络. 哈尔滨工业大学学报, 35(7): 823 - 825

侯云鹤,熊信艮. 2003. 基于广义蚁群算法的电力系统经济负荷分配. 中国电机工程学报, 23(3): 59 - 64

蒋建国,骆正虎. 2003. 基于改进型蚁群算法求解旅行 Agent 问题. 模式识别与人工智能, 16(1): 6 - 11

李艳君,吴铁军. 2001. 连续空间优化问题的自适应蚁群系统算法. 模式识

别与人工智能，14(4)：423－427

李勇，段正澄. 2003. 动态蚁群算法求解 TSP 问题. 计算机工程与应用，39(17)：103－106

吕勇，赵光宙. 2003. 蚁群优化算法及其在电力系统中的应用. 电工技术学报，18(4)：70－74

王恒奎，边耐欣. 2002. 基于 Trimmed NURBS 曲面几何特征的数字化自适应采样. 计量学报，23(4)：271－275

汪镭，吴启迪. 2003. 蚁群算法在系统辨识中的应用. 自动化学报，29(1)：102－109

汪镭，吴启迪. 2003. 蚁群算法在连续空间寻优问题求解中的应用. 控制与决策，18(1)：45－48，57

王笑蓉，吴铁军. 2003. Flowshop 问题的蚁群优化调度方法. 系统工程理论与实践，23(5)：65－71

王颖，谢剑英. 2002. 一种基于蚁群算法的多媒体网络多播路由算法. 上海交通大学学报，36(4)：526－528

吴庆洪，张纪会，徐心和. 1999. 具有变异特征的蚁群算法. 计算机研究与发展，36(10)

熊伟清，余舜洁. 2003. 具有分工的蚁群算法及应用. 模式识别与人工智能，16(3)：328－333

徐宁，朱小科. 2002. 用于两端线网布线的蚁群系统方法. 计算机辅助设计与图像学学报，14(5)：1－3

杨勇，宋晓峰. 2003. 蚁群算法求解连续空间优化问题. 控制与决策，18(5)：573－576

游道明，陈坚. 2003. 用蚂蚁算法解决多目标 TSP 问题. 小型微型计算机系统，24(10)：1808－1811

张国钢，耿英三. 2003. 基于蚁群并行算法的电气接线路径优化及仿真. 系统仿真学报，15(8)：1901－1904

张素兵，刘泽民. 2001. ATM 业务控制中的一种新的神经网络方法. 北京邮电大学学报，24(2)：15－19

张宗永，孙静，谭家华. 2002. 蚁群算法的改进及其应用. 上海交通大学学报，36(11)：1564－1567

周伟,刘粉林. 2003. 简单蚁群算法的仿真分析. 控制与决策, 18(3): 317 - 319

Abbaspour K C, Schulin R. 2001. Estimating unsaturated soil hydraulic parameters using ant colony optimization. *Advances in Water Resources*, Aug., 24(8): 827 - 841

Abbattista F, Abbattista N, Caponetti L. 1995. An evolutionary and cooperative agents model for optimization. *IEEE International Conference on Evolutionary Computation*, 2: 668 - 671

Akdaqli A, Guney K. 2002. Pattern nulling of linear antenna arrays by controlling only the element positions with the use of improved touring ant colony optimization algorithm. *Journal of Electromagnetic Waves and Applications*, 16(10): 1423 - 1441

Almulla M, Szuba T. 1999. Toward a computational model of collective intelligence and its IQ measure. *Proc. of the 14th ACM Symposium on Applied Computing*: 2 - 7

Bay J S. 1995. Design of the "army-ant" cooperative lifting robot. *IEEE Robotics & Automation Magazine*, Mar., 2(1): 36 - 43

Bilchev G, Parmee I C. 1995. Searching heavily constrained design spaces. *Proc. of 22nd International Conference on Computer Aided Design*: 230 - 235

Bilchev G, Parmee I C. 1995. The ant colony metaphor for searching continuous design spaces. *Evolutionary Computing, AISB Workshop Selected Papers*: 25 - 39

Bland J A. 1999. Layout of facilities using an ant system approach. *Engineering Optimization*, 32(1):101 - 115

Bland J A. 1999. Space-planning by ant colony optimization. *International Journal of Computer Applications in Technology*, 12(6): 320 - 328

Boryczka M. 1998. Some aspects of ant systems for the TSP. *Fundamenta Information*, Aug., 35(1 - 4): 197 - 209

Boryczka U, Boryczka M. 1998. Generative policies in ant systems for scheduling. *6th European Congress on Intelligent Techniques and Soft Computing*, 1: 382 - 386

Boryczka U. 1998. Learning with delayed rewards in ant systems for the job-shop scheduling problem. *First International Conference on Rough Sets and Current Trends in Computing*: 271 - 274

Boryczka U. 1999. Bus routing problem and the ant colony system. *Computational Intelligence for Modeling ,Control and Automation Congress*, Feb. : 11 - 16

Broqqi A, Cellario M. 2003. An evolutionary approach to visual sensing for vehicle navigation. *IEEE Transactions on Industrial Electronics*, Feb. , 50(1): 18 - 29

Bullnheimer B, Hartl R F, Strauss C. 1999. An improved ant system algorithm for the vehicle routing problem. *Annals of Operations Research*,89: 319 - 328

Bullnheimer B, Hartl R F, Strauss C. 1999. A new rank based version of the ant system—a computational study. *Central European J. of Operations Research*, (1): 25 - 38

Cartwright H M, Hopkins J A. 1996. Evolutionary design of synthetic routes in chemistry. *Evolutionary Computing, AISB Workshop Selected Papers*: 23 - 38

Chang C S, Tian L. 1999. New approach to fault section estimation in power systems using Ant system. *Electric Power Systems Research*, Feb. , 49(1): 63 - 70

Chen Ling, Shen Jie. 2002. Method for solving optimization problem in continuous space by ant colony algorithm. *Jounal of Software*, Dec. , 13(12): 2317 - 2323

Chen Wei-Shing, Chyu Chiuh-Cheng. 2002. Reduction of printed circuit board group assembly time through the consideration of efficiency loss of placement time. *Assembly Automation*, 22(4): 360 - 370

Chen Yibao, Yao Jianchu. Ant system based optimization algorithm and its applications in identical parallel machine scheduling. *Journal of Systems Engineering and Electronics*, Sep. , 13(3): 78 - 85

Chou C S, Song Y H. 1997. Ant colony-tabu approach for combined heat and power economic dispatch. *Proc. of the 1997 32nd Universities Power Engineering Conference*, Part 2: 605 - 608

Coello C A C, Zavala G. 2002. Automated design of combinational logic circuits

using the ant system. *Engineering Optimization*, Mar., 34(2):109 - 127

Coello C A C. 2002. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, Jan., 191(11 - 12): 1245 - 1287

Collett T S. 1998. How are studies of insect navigation useful to robotocists? *Proc. of the 1998 IEE Colloquium on Self-learning Robots* Ⅱ: *Bio-Robotics*: 8

Colorni A, Dorigo M, Maffioli F, Maniezzo V, Righini G, Trubian M. 1996. Heuristics from nature for hard combinatorial optimization problems. *International Trans. In Operational Research*, 3(1): 1 - 21

Costa D, Hertz A. 1997. Ants can colour graphs. *Journal of the Operational Research Society*, Mar., 48(3): 295 - 305

De Campos Luis M, Fernandez-Luna Juan M. 2002. Ant colony optimization for learning Bayesian networks. *International Journal of Approximate Reasoning*, Nov., 31(3): 291 - 311

Dorigo M, Maniezzo V, Colorni A. 1994. Introduction to natural algorithms. *Rivista-di-Informatica*, 24(3): 179 - 197

Dorigo M, Gambardella L M. 1996. A study of some properties of Ant-Q. *International Conference on Evolutionary Computation*, Sept.: 656 - 665

Dorigo M, Maniezzo V. 1996. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Mans, and Cybernetics, Part B: Cybernetics*, Feb., 26(1): 29 - 41

Dorigo M, Gambardella L M. 1997. Ant colony system: A cooperative learning approach to the traveling saleman problem. *IEEE Transactions on Evolutionary Computation*, Apr., 1(1): 53 - 66

Dorigo M, Bonabeau E. 2000. Ant algorithms and stigmergy. *Future Generation Computer Systems*, June, 16(8): 851 - 871

Douzono H, Hara S, Kawamoto S, Noguchi Y. 1998. A clustering method using genetic algorithm and ant system. *Joint Conference on Intelligent Systems*, 2: 407 - 410

Eqqers J, Feillet D. 2003. Optimization of the keyboard arrangement problem

using an Ant Colony algorithm. *European Journal of Operational Research*, Aug., 148(3): 672 - 686

Fukuda T, Mizoguchi H, Sekiyama K, Arai F. 1999. Group behavior control for Micro Autonomous Robotic System. *Proc. of the 1999 IEEE International Conference on Robotics and Automation*, 2: 1550 - 1555

Fulkerson B. 1995. Can ants aid manufacturing? *Resource: Engineering & Technology for Sustainable World*, Dec., 2(12): 2

Gagne C, Gravel M. 2002. Comparing an ACO algorithm with other heuristics for the single machine scheduling problem with sequence-dependent setup times. *Journal of the Operational Research Society*, Aug., 53(8): 859 - 906

Gagne C, Gravel M. 2002. Ant colony optimization algorithm with multiple visibility matrices for the resolution of a problem of industrial plant scheduling. *INFOR Journal*, Aug., 40(3): 259 - 276

Gambardella L M, Dorigo M. 1996. Solving symmetric and asymmetric TSPs by ant colonies. *Proc. of the 1996 IEEE International Conference on Evolutionary Computation*: 622 - 627

Gambardella L M, Taillard E D, Dorigo M. Ant colonies for the quadratic assignment problem. *J. of the Operational Research Society*, 50(2): 167 - 176

Gambardella L M, Dorigo M. 1995. Ant-Q: a reinforcement learning approach to the travelling salesman problem. *Proc. of the 12th International Conference on Machine Learning*: 252 - 260

Gamez J A, Puerta J M. 2002. Searching for the best elimination sequence in Bayesian networks by using ant colony optimization. *Pattern Recognition Letters*, Jan., 23(1 - 3):261 - 277

Gao Wei, Zheng Yingren. 2002. Ant colony algorithm and its application into optimization of construction order for underground house groups. *Chinese Journal of Rock Mechanics and Engineering*, Apr., 21(4): 471 - 474

Geurts F. 1998. ANTS98—from ant colonies to artificial ants: First International Workshop on Ant Colony Optimization. *AI Communications*, 11(3): 241 - 242

Gravel M, Price W L. 2002. Scheduling continuous casting of aluminum using a multiple objective ant colony optimization metaheuristic. *European Journal of*

Operational Research, Nov., 143(1): 218 - 229

Gutjahr W J. 2000. Graph-based Ant System and its convergence. *Future Generation Computer Systems*, June, 16(8): 873 - 888

Gutjahr W J. 2002. ACO algorithms with guaranteed convergence to the optimal solution. *Information Processing Letters*, May, 82(3):145 - 153

Heck P S, Ghosh S. 2000. Study of synthetic creativity: behavior modeling and simulation of an ant colony. *IEEE Intelligent Systems and Their Applications*, Nov., 15(6):58 - 66

Heck P S, Ghosh S. 2002. The design and role of synthetic creative traits in artificial ant colonies. *Theory and Application*, Apr., 33(4):343 - 370

Hirasawa K, Okubo M. 2001. Comparison between genetic network programming and genetic programming using evolution of ant's behaviors. *Research Reports on Information Science and Electrical Engineering of Kyushu University*, Mar., 6(1): 31 - 37

Hiura A, Kuroda T, Inuzuka N, Itoh K, Yamada M, Seki H, Itoh H. 1997. Cooperative behavior of various agents in dynamic environment. *Computers & Industrial Engineering*, 33(3 - 4): 601 - 604

Hoshyar R, Jamali S H, Locus C. Ant colony algorithm for finding good interleaving pattern in turbo codes. *IEE Proc-Commun*, 147(5): 257 - 262

Huang S J. 2001. Enhangcement of hydroelectric generation scheduling using ant colony system based optimization approaches. *IEEE Transactions on Energy Conversion*, Sep., 16(3):296 - 301

Janson S, Merkle D. 2003. On Enforced Convergence of ACO and its Implementation on the Reconfigurable Mesh Architecture Using Size Reduction Tasks. *Journal of Supercomputing*, Nov., 26(3):221 - 238

Jayaraman V K, Kulkarni B D. 2000. Ant colony framework for optimal design and scheduling of batch plants. *Computers and Chemical Engineering*, Sep., 24(8):1901 - 1912

Jennings B, Brennan R. 1999. FIPA—compliant agents for real-time control of Intelligent Network traffic. *Computer Networks*, Aug., 31(19): 2017 - 2036

Karaboqa N, Guney K. 2002. Null steering of linear antenna arrays with use of

modified touring ant colony optimization algorithm. *International Journal of RF and Microwave Computer-Aided Engineering*, July, 12(4): 375 - 383

Kawamura H, Yamamoto M. 1998. Cooperative search based on pheromone communication for vehicle routing problems. *IEICE Tranctions on Fundamentals of Electronics, Communications and Computer Sciences*, June, E81 - A(6): 1089 - 1096

Kawamura H, Yamamoto M. 2000. Multiple ant colonies algorithm based on colony level interactions. *IEICE Transactions on Fundamentals of Electrons, Communications and Computer Sciences*, E83 - A(2):371 - 379

Keinprasit R, Chonqstitvatana P. 2003. High-Level Synthesis by Ants on a Tree. *IEICE Transactions on Fundamental of Electronics, Communications and Computer Sciences*, Oct., E86 - A(10):2659 - 2669

Keinprasit R, Chonqstitvatana P. 2004. High-level synthesis by dynamic ant. *International Journal of Intelligent Systems*, Jan. /Feb., 19(1 - 2):25 - 38

Kongmunvattana A, Chongstitvatana P. 1998. FPGA-based behavioral control system for a mobile robot. *Proc. of the 1998 IEEE Asia-Pacific Conference on Circuits and Systems*: 759 - 762

Kube C R, Bonabeau E. 2000. Cooperative transport by ants and robots. *Robotics and Autonomous Systems*, Jan., 30(1): 85 - 101

Kubo M, Kakazu Y. 1993. Simulating a competition for foods between ant colonies as a coordinated model of autonomous agents. *Proc. of the 1993 IEEE International Conference on SMC*, 5: 142 - 148

Kumar R, Tiwari M K. 2003. Scheduling of flexible manufacturing systems: An ant colony optimization approach. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 217(10): 1443 - 1453

Kuntz P, Snyers D, Layzell P. 1999. Stochastic heuristic for visualizing graph clusters in a bi-dimensional space prior to partitioning. *J. of Heuristics*, 5(3): 327 - 351

Kwang Mong Sim, Weng Hong Sun. 2003. Ant Colony Optimization for Routing and Load-Balancing: Survey and New Directions. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 33(5): 560 - 572

Lambrinos D, Moller R. 2000. Mobile robot employing insect strategies for navigation. *Robotics and Autonomous Systems*, Jan., 30(1): 39 - 64

Leguizamon G, Michalewicz Z. 1999. a new version of ant system for subset problems. *Proc. of the 1999 Congress on Evolutionary Computation*, 2: 1459 - 1464

Li Yan-Jun, Wu Tie-jun. 2003. Adaptive ant colony algorithm for continuous-space optimization problems. *Journal of Zhejiang University*, Jan., 4(1): 40 - 46

Li Yong, Gong Shihua. 2003. Dynamic ant colony optimization for TSP. *International Journal of Advanced Manufacturing Technology*, 22(7 - 8):528 - 533

Maier H R, Simpson A R. 2003. Ant colony optimization for design of water distribution systems. *Journal of Water Resources Planning and Management*, May/June, 129(3):200 - 209

Maniezzo V, Dorigo M, Colorni A, Algodesk. 1995. An experimental comparison of eight evolutionary heuristics applied to the Quadratic Assignment Problem. *European J. of Operational Research*, 81(1): 188 - 204

Maniezzo V. 1999. Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem. *Informs J. on Computing*, 11(4): 358 - 369

Maniezzo V, Colorni A. 1999. Ant systems applied to the quadratic assignment problem. *IEEE Transactions on Knowledge and Data Engineering*, Sep., 11(5): 769 - 778

Maniezzo V, Carbonaro A. 2000. ANTS heuristic for the frequency assignment problem. *Future Generation Computer Systems*, June, 16(8):927 - 935

Mathur M, Karale S B. 2000. Ant colony approach to continuous function optimization. *Industrial and Engineering Chemistry Research*, Oct., 39(10):3814 - 3822

McMullen P R. 2001. An ant colony optimization approach to addressing a JIT sequencing problem with multiple objectives. *Artificial Intelligence in Engineering*, July, 15(3): 309 - 317

Merkle D, Middendorf M. 2002. Ant colony optimization for resource-constrained project scheduling. *IEEE Transactions on Evolutionary Computation*, Aug.,

6(4): 333 - 346

Merkle D, Middendorf M. 2003. Ant colony optimization with global pheromone evaluation for scheduling a single machine. *Applied Intelligence*, Jan./Feb., 18(1):105 - 111

Michel R, Middendorf M. 1998. An island model based ant system with look-ahead for the shortest super sequence problem. *Proc. of the 5th International Conference on Parallel Problem Solving from Nature*: 692 - 701

Middendorf M, Reischle F. 2002. Multi colony ant algorithms. *Journal of Heuristics*, May, 8(3):305 - 320

Monmarche N, Venturini G. 2000. On how Pachycondyla apicalis ants suggest a new search algorithm. *Future Generation Computer Systems*, June, 16(8): 937 - 946

Mostefai N, Bourjault A. 1997. Modeling ethology inspired scenarios for mobile mini-robotics using object-oriented Petri-nets. *Proceedings of the 1997 IEEE 6th International Conference on Emerging Technologies and Factory Automation*: 431 - 435

Nemes L, Roska T. 1995. CNN model of oscillation and chaos in ant colonies: A case study. *IEEE Transactions on Circuits and Systems 1: Fundamental Theory and Applications*, Oct., 42(10):741 - 745

Nowe A, Verbeeck K. 1999. Formalizing the ant algorithms in terms of reinforcement learning. *Proc. of the 5th European Conference in Artificial Life*: 616 - 620

Ozturk S, Esin E M. 2003. Simulation of swarm intelligence and possible applications in engineering. *Mathematical and Computational Applications*, 2(1 - 3): 361 - 368

Parmee I C, Vekeria H. 1997. Role of evolutionary and adaptive search during whole system, constrained and detailed design optimization. *Engineering Optimization*, 29(1 - 4): 151 - 176

Parmee I C. 1998. Evolutionary and adaptive strategies for efficient search across whole system engineering design hierarchies. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM*, Nov., 12(5): 431 - 445

Parpinelli R S, Lopes H S. 2002. Data mining with an ant colony optimization al-

gorithm. *IEEE Transactions on Evolutionary Computation*, Aug., 6(4):321 - 332

Rajendran C, Zieqler H. 2004. Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs. *European Journal of Operational Research*, June, 1, 155(2):426 - 438

Rajesh J, Gupta K. 2001. Dynamic optimization of chemical processes using ant colony framework. *Computers and Chemistry*, Nov., 25(6):583 - 595

Reimann M, Doerner K. 2004. D-ants: Savings based ants divide and conquer the vehicle routing prolem. *Computers and Operations Research*, Apr., 31(4): 563 - 591

Ronald K C, Zhang H. 1994. Stagnation recovery behaviors for collective robotics. *Proceedings of the IEEE International Conference on Intelligent Robotics and Systems*, 3: 1883 - 1890

Russell R A. 1999. Ant trails—an example for robots to follow? *IEEE International Conference on Robotics and Automation*, 4: 2698 - 2703

Russell R A. 1998. Odour sensing robot draws inspiration from the insect world. *Proc. of the 1998 2nd International Conference on Bioelectromagnetism*: 49 - 50

Sandalidis H G, Mavromoustakis C X. 2004. Ant based probabilistic routing with pheromone and antipheromone mechanisms. *International Journal of Communication Systems*, Feb., 17(1): 55 - 62

Schoonderwoerd R, Holland O, Bruten J. 1997. Ant-like agents for load balancing in telecommunications networks. *Proc. of the First International Conference on Autonomous Agents*: 209 - 216

Schoonderwoerd R, Holland O, Bruten J, Rosenkrants L. 1996. Ant-based load balancing in telecommunications networks. *HP Laboratories Technical Report*, 26:96 - 76

Sharman E K, Unsal C, Bay J S. 1997. Areactive coordination scheme for a many-robot system. *IEEE Trans. On SMC Part B: Cybernetics*, 27(4): 598 - 610

Shelokar P S, Jayaraman V K. 2002. Ant algorithm for single and multiobjective reliability optimization problems. *Quality and Reliability Engineering Interna-*

tional, Nov./Dec., 18(6): 497 - 514

Shelokar P S, Jayaraman V K. 2003. Multiobjective optimization of reactor-regenerator system using ant algorithm. *Petroleum Science and Technology*, July/Aug., 21(7 - 8):1167 - 1184

Shouse B. 2002. Getting the behavior of social insects to compute. *Science*, Mar., 295(5564): 2357

Shimoyama I, Miura H, Kimura C, Kikuta M. 1991. Analyzing the dynamics of ants and application to micro-robots. *Winter Annual Meeting of the American Society of Mechanical Engineering*: 279 - 285

Siqel E, Denby B. 2002. Application of ant colony optimization to adaptive routing in a LEO telecommunications satellite network. *Annales des Telecommunications/Annuals of Telecommunications*, May/June, 57(5 - 6):520 - 539

Solnon C. 2002. Ants can solve constraint satisfaction problems. *IEEE Transactions on Evolutionary Computation*, Aug., 6(4): 347 - 357

Song Y H, Chou C S. 1999. Combined heat and power economic dispatch by improved ant colony search algorithm. *Electric Power Systems Research*, Nov., 52(2):115 - 121

Song Y H, Chou C S. 1999. Large-scale economic dispatch by artificial ant colony search algorithms. *Electric Machines and Power Systems*, July, 27(7): 679 - 690

Stilwell D J, Bay J S. 1993. Toward the development of a material transport system using swarms of ant-like robots. *Proceedings of the IEEE International Conference on Robotics and Automation*: 766 - 771

Stutzle T, Hoos H. 1997. Max-Min ant system and local search for the travelling salesman problem. *Proc. of* 1997 *IEEE International Conference on Evolutionary Computation*: 309 - 314

Stutzle T, Hoos H. 2000. MAX-MIN Ant System. *Future Generation Computer Systems*, June, 16(8):889 - 914

Stutzle T, Dorigo M. 2002. A short convergence proof for a class of ant colony optimization algorithms. *IEEE Transactions on Evolutionary Computation*, Aug., 6(4): 358 - 365

Subrata R, Zomaya A Y. 2003. A comparison of three artificial life techniques for reporting cell planning in mobile computing. *IEEE Transactions on Parallel and Distributed Systems*, Feb., 14(2): 142 - 153

Sum J, Shen H. 2003. Analysis on a mobile agent-based algorithm for network routing and management. *IEEE Transactions on Parallel and Distributed Systems*, Mar., 14(3): 193 - 200

Sum J, Shen H. 2003. Analysis on extended ant routing algorithms for network routing and management. *Journal of Supercomputing*, Mar., 24(3): 327 - 340

Sun Ruoying, Tatsumi Shoji. 2003. Multiagent Cooperating Learning Methods by Indirect Media Communication. *IEICE Transactions on Fundamentals of Electrons, Communications and Computer Sciences*, Nov., E86 - A(11): 2868 - 2878

Sun Zhi-Guo, Teng Hong-Fei. 2003. Optimal layout design of a satellite module. *Engineering Optimization*, Oct., 35(5): 513 - 529

Taillard E D, Gambardella L M. 2001. Adaptive memory programming: A unified view of metaheuristics. *European Journal of Operational Research*, Nov., 135(1): 1 - 16

Talbi E G, Roux O. 2001. Parallel Ant Colonies for the quadratic assignment problem. *Future Generation Computer Systems*, Jan., 17(4):441 - 449

Teng Jen-Hao, Liu Yi-Hwa. 2003. A novel ACS-based optimum switch relocation method. *IEEE Transaction on Power Systems*, Feb., 18(1): 113 - 120

Teodorovic D. 2003. Transport modeling by multi-agent systems: A swarm intelligence approach. *Transportation Planning and Technology*, Aug., 26(4): 289 - 312

Tkindt V, Monmarche N. 2002. An Ant Colony Optimization algorithm to solve a 2-machine bicriteria flowshop scheduling problem. *European Journal of Operational Research*, Oct., 142(2): 250 - 257

Tsai Cheng-Fa, Tsai Chun-Wei. 2003. A new and efficient ant-based heuristic method for solving the traveling salesman problem. *Expert Systems*, Sep., 20(4):179 - 186

Valckenaers P, Hendrik V B, Wyns J, Patrick P, Bongaerts L. 1999. Multi-

agent manufacturing control in holonic manufacturing systems. *Human Systems Management*, 18(3): 233 - 243

Valckenaers P, Kollingbaum M. 2004. Multi-agent coordination and control using stigmergy. *Computers in Industry*, Jan., 53(1): 75 - 96

Verbeeck K, Nowe A. 2002. Colonies of learning automata. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, Dec., 32(6): 772 - 780

Vijayakumar K, Prabhaharan G. 2003. Optimization of multi-pass turning operations using ant colony system. *International Journal of Machine Tools and Manufacture*, Dec., 43(15): 1633 - 1639

Vittori K, Araujo A F R. 2001. Agent-oriented routing in telecommunications networks. *IEICE Transactions on Communications*, Nov., E84 - B(11): 3006 - 3013

Voth D. 2002. Nature's guide to robot design. *IEEE Intelligence Systems*, Nov./Dec., 17(6): 4 - 7

Wagner I A, Lindenbaum M. 1999. Distributed covering by ant-robots using evaporating traces. *IEEE Transactions on Robotics and Automation*, Oct., 15(5): 918 - 933

Wang Chunfeng, Zhao Xin. 2002. Ants foraging mechanism in the design of multi-product batch chemical process. *Industrial and Engineering Chemistry Research*, Dec., 41(26): 6678 - 6686

Wang Lei, Wang Xiaoping. 2002. Ant System Algorithm based Rosenbrock Function Optimization in Multi-dimension Space. *First International Conference on Machine Learning and Cybernetics*, Nov., (4 - 5)

Wang Lei, Wu Qidi. 2001. Ant System Algorithm for Optimization in Continuous Space. *Proceedings of IEEE CCA/ISIC Conference*, Sept., (5 - 7): 395 - 399

Wang Lei, Wu Qidi. 2001. Linear System Parameters Identification based on Ant System Algorithm. *Proceedings of IEEE CCA/ISIC Conference*, Sept., (5 - 7): 401 - 406

Wang Lei, Wu Qidi. 2002. Ant System Algorithm Research and its Applications. *High Technology Letters*, 8(4): 91 - 96

Yanovski Vladimir, Wagner Israel A. 2003. A distributed ant algorithm for

efficiently patrolling a network. *Algorithmical (New York)*, Nov., 37(3): 165 - 186

Yin Peng-Yeng. 2003. Ant colony search algorithms for optimal polygonal approximation of plane curves. *Pattern Recognition*, Aug., 36(8): 1783 - 1797

Ying Kuo-Chinq, Liao Ching-Jong. 2003. An ant colony system approach for scheduling problems. *Production Planning and Control*, Jan./Feb., 14(1): 68 - 75

Ying Kuo-Chinq, Liao Ching-Jong. 2004. An ant colony system for permutation flow-shop sequencing. *Computers and Operations Research*, Apr., 31(5): 791 - 801

Yu I K, Song Y H. 2001. A novel short-term generation scheduling technique of thermal units using ant colony search algorithms. *International Journal of Electrical Power and Energy System*, Aug., 23(6): 471 - 479

Yun C L, Smith A E. 1999. An ant system approach to redundancy allocation. *Proc. of the 1999 Congress on Evolutionary Computation*, 2: 1478 - 1484

Zhuang C W, Fan M Y, Li C H, Yu J B. 1999. Ant colony switchbox router based on coordination mechanism. *Chinese J. of Semiconductors*, 20(5): 400 - 406

[General Information]
书名=智能蚁群算法及应用
作者=吴启迪　汪镭著
页数=213
SS号=11402640
DX号=
出版日期=2004年04月第1版
出版社=上海科技教育出版社